NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

DYNAMIC ALLOCATION OF FIRES AND SENSORS

by

Michael E. Havens

September 2002

Thesis Advisor: Arnold Buss
Second Reader: Matthew Chesney

Approved for public release; distribution is unlimited.

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			*Form Approved OMB No. 0704-0188*
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.			
1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** September 2002	**3. REPORT TYPE AND DATES COVERED** Master's Thesis	
4. TITLE AND SUBTITLE: Title (Mix case letters) Dynamic Allocation of Fires and Sensors			**5. FUNDING NUMBERS**
6. AUTHOR(S) Michael E. Havens			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000			**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A			**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.			**12b. DISTRIBUTION CODE** A

13. ABSTRACT *(maximum 200 words)*

The U.S. Army is undergoing significant changes in its force structure and implementation doctrine. This thesis evaluates factors associated with networking assets in a future battle space incorporating Future Combat Systems. An analysis framework was developed designed to assist the Army in current and future evaluation of networked assets and potential configurations of Future Combat Systems at the Unit of Action (UA) and Entity levels. The framework consists of a Discrete Event Simulation Model, Extensible Mark-up Language (XML) input and output modules, and an output analysis package. The simulation model receives scenario inputs from XML files. During the simulation run, the model intermittently calls an optimization package that solves a multi-dimensional knapsack problem to allocate assets based on the current conditions. Once the simulation is complete the model generates XML output that is subsequently processed by an analysis package. The model goes beyond normal implementations of both simulation and optimization by incorporating both simultaneously. The result is an increased level of analysis quality due to the consideration of both stochastic factors and optimization techniques and an analysis architecture that will serve the Army as a basis for the exploration of factors associated with networking assets and system configurations.

14. SUBJECT TERMS Networked Fires and Sensors, U.S Army Future Combat Systems, Objective Force, Discrete Event Simulation, Mathematical Programming, Optimization, Extensible Mark-up Language (XML), Knapsack problem.			**15. NUMBER OF PAGES** 97
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited.

DYNAMIC ALLOCATION OF FIRES AND SENSORS

Michael E. Havens
Lieutenant, United States Navy
B.S.M.E., The Ohio State University, 1994

Submitted in partial fulfillment of the
Requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
September 2002

Author: Michael E. Havens

Approved by: Arnold Buss
Thesis Advisor

MAJ Matt Chesney, USA
Second Reader

James N. Eagle, Chairman
Department of Operations Research

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

The U.S. Army is undergoing significant changes in its force structure and implementation doctrine. This thesis evaluates factors associated with networking assets in a future battle space incorporating Future Combat Systems. An analysis framework was developed designed to assist the Army in current and future evaluation of networked assets and potential configurations of Future Combat Systems at the Unit of Action (UA) and Entity levels. The framework consists of a Discrete Event Simulation Model, Extensible Mark-up Language (XML) input and output modules, and an output analysis package. The simulation model receives scenario inputs from XML files. During the simulation run, the model intermittently calls an optimization package that solves a multi-dimensional knapsack problem to allocate assets based on the current conditions. Once the simulation is complete the model generates XML output that is subsequently processed by an analysis package. The model goes beyond normal implementations of both simulation and optimization by incorporating both simultaneously. The result is an increased level of analysis quality due to the consideration of both stochastic factors and optimization techniques and an analysis architecture that will serve the Army as a basis for the exploration of factors associated with networking assets and system configurations.

THIS PAGE INTENTIONALLY LEFT BLANK

DISCLAIMER

The reader is cautioned that computer programs developed in this research may not have been exercised for all cases of interest. While every effort has been made, within the time available, to ensure that the programs are free of computational errors, they cannot be considered validated. Any application of these programs without additional verification is at the risk of the planner.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

During the course of this thesis process many individuals have been available to assist, motivate and encourage. To all those who have, I give my thanks and appreciation.

First and foremost I thank God for providing me the talents, persistence and opportunity to be able to pursue this task. To my wife and best friend Sabrina, thank you for your patience, support and the motivation to pursue excellence; I love you with all I am. Ashly, Michel, Jacob and Blake, thanks for being supportive and listening to me talk about all this crazy stuff.

Professor Arnold Buss, thanks for keeping me on the path and for the countless lessons you have provided. The experience of working with you has been an education itself. Matt Chesney and Jack Jackson, thanks for the technical guidance and the willingness to help pull a Navy guy through an army topic.

I also thank the members of the NPS Operations Research faculty and my fellow students. Though you didn't always know it, many of you were there at just the right time to offer re-enforcement and encouragement; I respect and appreciate you all.

THIS PAGE INTENTIONALLY LEFT BLANK

EXECUTIVE SUMMARY

The U.S. Army is currently fully engaged in a significant transformation effort that will allow them to meet all mission requirements and be a truly effective element in joint operations during the first quarter of this century. A cornerstone in the transformation is the development and deployment of a land-based Objective Force that is flexible and capable enough to meet all challenges to future national security. In order to accomplish this, the Army has stated that the Objective Force must be capable of full spectrum operations, strategically mobile, lethal, and survivable and must take advantage of new technologies to the fullest extent.

The Objective Force departs from traditional Army force structure through the insertion of medium-weight forces referred to as Future Combat Systems (FCS). The FCS will be far more strategically mobile than the current heavy systems and provide more lethality than current light forces. The idea is to bring to bear a balance between the two that will provide increased capability sooner. In the case where heavy forces would eventually be required, the FCS force will provide a more capable standoff during the deployment phase. Conversely, the FCS force may be able to prevent the need for heavy forces altogether by being sufficiently effective in smaller-scale contingencies.

The FCS will function as a system of systems with its survivability and lethality increases relying heavily on information fusion in the battle space. During the next two decades, it is unlikely that progresses in technology will allow full battle space visibility from aerial or other sensors. Therefore, the Army must continue to evaluate means to fuse battle space information from all sources and utilize it effectively. The effective utilization of this fused information refers to networked fires: the concept of allocating fires within the battle space as a result of applied information. The combination of effective use of networked fires and the specific configurations of FCS units and Units of Action UA, are key developmental issues facing the Army in its effort to field the Objective Force.

This thesis is directed at providing the Army with two distinct products; first, the initial infrastructure and conceptual framework for an analysis tool that allows the

combined exploration of UA configurations and networked fires logic and second, initial analysis results from the model. Because networked fires logic and FCS configurations are directly related to one another, the ability to study both simultaneously is critical. In this thesis, both are addressed through the use of a robust simulation architecture that implements an optimization routine. Both the simulation and the optimization components are designed to allow maximum flexibility with regard to structure and implementation.

The Dynamic Allocation of Fires and Sensors (DAFS) model departs from normal implementations of simulation and optimization by applying both simultaneously. The framework developed utilizes Extensible Mark-up Language (XML) to a great extent in order to provide DAFS with ease and flexibility in scenario development and output analysis.

The DAFS model implements a discrete event simulation that intermittently calls an optimization solver package. The optimization problem solved is a linear integer multi-dimensional knapsack problem with generalized set packing and set covering constraints. Together the simulation and optimization work together to try and achieve as near an optimal solution to a unique engagement as possible. The use of modeled sensors, firers and the allocation optimization allows DAFS to dynamically adjust to the battle space situation in real time in an attempt to maximize success.

In this thesis the model was used to evaluate initial factors associated with success of networking fires in a meeting engagement. The engagement involved elements of a battalion UA for blue versus elements of a red brigade. Through the use of modeled sensors, the battle space situation was developed on the fly and the blue firing units were allocated as a result of the intermittent optimizations. Additionally, sensor units were allocated for battle damage assessment (BDA) based on firing unit reports of engagement. The BDA sensors were also allocated as a result of an optimization.

The analysis conducted indicates that the tactical values, level of BDA accuracy and the optimization interval are all significant to unit survivability on both sides. Additionally, the results indicate that the blue survivability is more robust to the range of factor settings and that policy based on these factors could be set with a more weight assigned to the consideration of red survivability.

Through this thesis, DAFS has been demonstrated to be an effective and flexible analysis framework for the evaluation of factors associated with networked assets and FCS configurations. Additionally, the component-based design of DAFS provides the potential user with extensive latitude in the definition of units and applied logic.

The results obtained in this thesis are merely representative of the potential of DAFS. The potential of DAFS in assisting the Army in the analysis of its time-critical consideration of FCS is high. Continued refinement of the DAFS model and application to the Army's research is highly recommended.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

The U.S. Army is currently fully engaged in a significant transformation effort that will allow them to meet all mission requirements and be a truly effective element in joint operations during the first quarter of this century. A cornerstone in the transformation is the development and deployment of a land-based Objective Force that is flexible and capable enough to meet all challenges to future national security. In order to accomplish this, the Army has stated that the Objective Force must be capable of full spectrum operations, strategically mobile, lethal, and survivable and must take advantage of new technologies to the fullest extent.

The Objective Force departs from traditional Army force structure through the insertion of medium-weight forces referred to as Future Combat Systems (FCS). The FCS will be far more strategically mobile than the current heavy systems and provide more lethality than current light forces. The idea is to bring to bear a balance between the two that will provide increased capability sooner. In the case where heavy forces would eventually be required, the FCS force will provide a more capable standoff during the deployment phase. Conversely, the FCS force may be able to prevent the need for heavy forces altogether by being sufficiently effective in smaller-scale contingencies. Figure 1Figure 1. demonstrates the perceived impact of the FCS.

The FCS will function as a system of systems with its survivability and lethality increases relying heavily on information fusion in the battle space. During the next two decades, it is unlikely that progresses in technology will allow full battle space visibility from aerial or other sensors. Therefore, the Army must continue to evaluate means to fuse battle space information from all sources and utilize it effectively. The effective utilization of this fused information refers to networked fires; the concept of allocating fires within the battle space as a result of applied information. The combination of effective use of networked fires and the specific configurations of FCS units and Units of Action UA, are key developmental issues facing the Army in its effort to field the Objective Force.

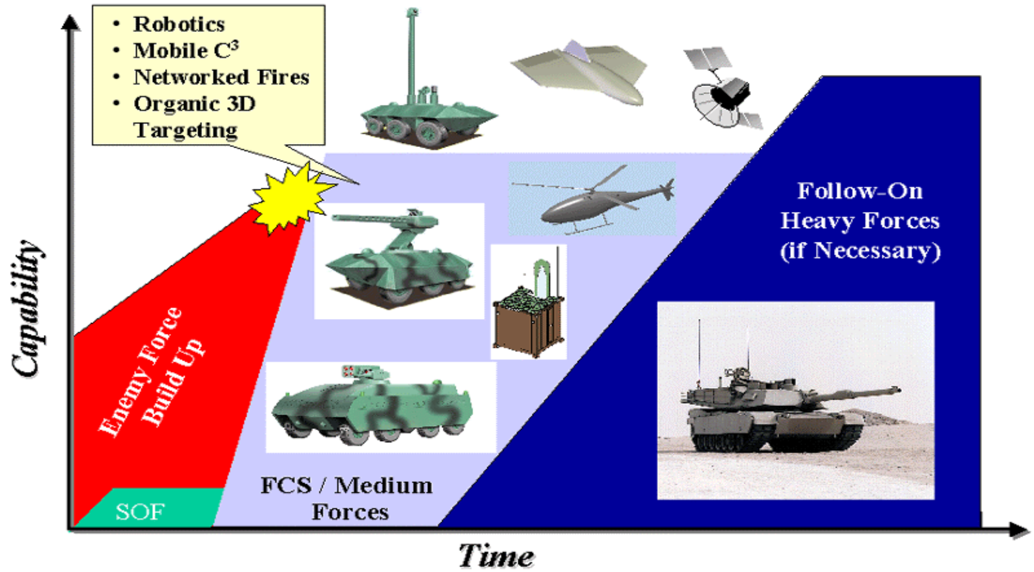


Figure 1. FCS concept[1]

In direct support of the Army's goals, the US Army Training and Doctrine Command (TRADOC) Analysis Center (TRAC), has undertaken an extended project directed at evaluating all critical factors associated with the development of FCS. TRAC-Monterey has been tasked with initial analysis of several of these factors. This thesis is a sponsored by TRAC-Monterey and focused on the analysis of networked fires and FCS unit configurations. .

A. RESEARCH OBJECTIVES

This thesis is directed at providing TRAC-Monterey and the Army with two distinct products; first, the initial infrastructure and conceptual framework for an analysis tool, a simulation model, that allows the combined exploration of UA configurations and networked fires logic and second, initial analysis results using the simulation model. Because networked fires logic and FCS configurations are directly related to one another, the ability to study both simultaneously is critical. In this thesis, both are addressed through the use of a robust simulation architecture that implements an optimization routine. Both the simulation and the optimization component are designed to allow maximum flexibility with regard to structure and implementation.

[1] Reprinted with permission. Objective Force Task Force presentation of June 2001.

1. Networked Fires

Networked fires refers to the application of fused battle space information that is gathered through various sensors, compiled and used as a collective tool for the future direction of specific units' fires within the battle space. The ability to effectively accomplish this applied fusion of information leads directly to all units in the battle space functioning as a system of systems, one of the stated goals of FCS development.

Networked fires addresses increases in both the effectiveness and the efficiency of an FCS force. While it has traditionally been the approach of the military to focus on overwhelming effectiveness, the current age of logistics and deployment time constraints have driven focus to the issue of efficiency. It is the potential improvement in overall force efficiency that makes the realization of networked fires a critical element in OF and FCS development.

This thesis expands the definition of networked fires to include the allocation of sensors as well. This is because the implementation of sensors in the modern battle space is often directly related to the ability to deliver fires. If the employment of all weapon systems and sensor capabilities can be managed together, the result may mean dramatic increases in the effectiveness and efficiency of the FCS force. The flexible nature of the model developed for this research enables the user to evaluate allocation of both fires and sensors.

2. FCS Evaluation

In addition to providing a means for the Army to evaluate the potential effectiveness of networked fires, the model can give insight into effective UA configurations that are currently in developmental stages. As additional proposals for UA configurations are presented, they may be analyzed in the same environment and conditions previously applied to other configurations.

The robust nature of the model developed allows analysis to be conducted using a wide range of perceived UA configurations and leads to inferences about the structure of the force as a whole and how to employ it. Every aspect of the FCS units being analyzed, as well as the environment, can easily be altered and further analysis conducted. Given the advanced nature of the overall effort to develop the OF and FCS, the ability to rapidly test the system definitions within an existing framework is of tremendous benefit to the Army.

3. Robust Simulation Architecture

The Dynamic Allocation of Fires and Sensors (DAFS) simulation framework is designed to provide maximum flexibility in the evaluation of networked fires for FCS. Through the use of interchangeable component based design, the simulation provides the user extensive ability to modify entities, configurations, simulation parameters and data output requirements.

DAFS is a discrete event simulation in JAVA that utilizes several components of Simkit; a JAVA simulation toolkit developed by Dr. Arnold Buss. In addition, several new classes were developed to extend the functionality of Simkit and meet the requirements of DAFS. The optimization routine in DAFS was configured using the open source JAVA version of a linear programming package called LP_Solve. Again, interface classes were developed to assist in the implementation of LP_Solve and enhance its benefit to DAFS. Finally, Extensible Mark-up Language (XML) is used for all inputs to DAFS. The use of XML for all inputs allows the user to conduct analysis by editing and applying simple XML files to define all aspects of the scenario desired. XML file information is imported to DAFS through the use of classes developed using JDOM[2] and SAX[3]; JAVA development software designed specifically for the purpose of reading XML files into JAVA code. Chapters III and IV present a detailed description of the DAFS model. The next chapter discusses the methodology underlying the model.

---

[2] JDOM is a JAVA API for XML document manipulation. JDOM is not an acronym but may thought of as a JAVA expansion of the W3C's Document Object Model (DOM) specification for XML document manipulation.

[3] Simple API for XML Parsing. SAX is an event-based support API for XML document parsing.

II. METHODOLOGY

As stated in the introduction, the goal of this research is to assist TRAC-Monterey in the analysis of the factors associated with the implementation of networked fires as well as potential configuration of FCS units. The problem, then, is to determine the critical elements involved in establishing the best way for a force, as a whole, to engage the opposing force it perceives itself to be up against. More specifically, what is the best way to dynamically allocate its assets in a successful manner?

This project approaches this problem through the synthesis of two operations research disciplines: simulation and optimization. Drawing on the benefit of optimization to provide an optimal solution to a static problem, and the ability of simulation to account for time varying and probabilistic factors, DAFS uses both to explore the factors associated with the use of networked fires and potential UA configurations.

A. PROBLEM DEFINITION

Given that a force could operate cooperatively with a networked backbone of information support, what are the critical factors in both the network, and force configuration, that would enable the highest rate of success against an opposing force with variable structures and tactics?

This question provides the basis for the model development. It is the critical factors that will enable the networked concept to be beneficial that must be determined if such a concept is to reach implementation. The following paragraphs describe the approach taken in this thesis using DAFS, and the associated spreadsheet tool (DAFS-ST) including the major assumptions.

1. Value of Potential Assignments (VPA)

One key premise behind the approach taken in the model development is the Value of Potential Assignments (VPA) in the battle space. VPA refers to the overall potential value of an assignment pairing between a friendly unit and a non-friendly unit.

This assignment is not necessarily a firing or sensing assignment though it may potentially lead to that end. Rather, it is a general assignment based on a number of potential factors such as engagement potential, tracking benefit, the overall threat the unit may present and many others. Placing a value on such an assignment is a necessary element and provides a means for analysis within the battle space. These factors turn out to be critical for analysis in networked fires.

The VPA concept takes into account several factors that contribute to assigning a particular friendly unit the responsibility of a particular non-friendly unit. As with the term assignment, responsibility is used here in a general sense to indicate focus of attention for a friendly unit. The idea is to take several potential factors available in the battle space that may influence a unit's actions, and process them in such a way that a final value or set of values is generated. Once this is done for each potential pairing of friendly to non-friendly units, those values are applied to an objective function designed to maximize the total value of a particular assignment set, based on a mission goal and a user-defined set of constraints[4].

a. VPA Factors

An Army unit considers many factors when considering whether or not it should engage or pursue an enemy unit. For the purposes of this thesis, a set of factors was chosen to capture the range of considerations while avoiding excessive detail. The general categories of factors chosen are probability of kill (P_k), threat (expressed as reverse probability of kill), inherent value of friendly and non-friendly units and the type of action engaged in (e.g. defense, peace keeping). While this may, at first glance, appear to be a very brief list of factors that would provide a limited factor space for exploration, indeed it is not. In each of these general areas there are extensive considerations and assumptions that may be made.

Some of the sub-factors related to the primary factors listed above are explored explicitly and some are explored implicitly and are presented in Table 1. The

[4] A minimization may also be chosen. The maximization terminology was chosen here merely to be consistent with the approach taken in this research.

implicit factors listed are influencing factors within the associated primary factor, which for the purposes of this research, are considered captured to a sufficient extent in the parent factor. Explicit modeling of these factors would cloud the process and provide an increased fidelity that is not necessary at this point in the development of DAFS.

Primary Factor	Explicit Sub-Factor	Implicit Sub-Factor
Probability of Kill (P_k)	Range Munitions Firing unit type Targeted unit type	Target Location Error (TLE) (Time-associated portion)[5] Munitions accuracy Munitions reliability
Threat	Range Firing enemy unit type Targeted friendly unit type	Munitions Munitions accuracy Munitions reliability
Unit values	Unit type Scenario type	Strategic value Monetary value
Action type	General category (attack, defend, etc…)	

Table 1. Explicit and Implicit Sub-factors[6]

b. VPA Use

The description and examples presented below are very simple representations of the concepts used in the following sections describing DAFS, DAFS-ST and experimental design points. They are merely intended to, in simple terms, demonstrate the logic template used as an approach to the project.

The use of the VPA and the associated formula used to arrive at it are the two main variables used to evaluate the potential benefits of networked fires. As will be discussed in subsequent sections, the VPA is generated as a result of a value formula that takes into account whatever factors have been designed into it. For example, one might propose that the factor involved in determining the VPA from a blue unit to a red unit is the expected value of eliminating the red unit. In this case, the VPA would be the red

[5] The target location accuracy is the other portion of TLE and is considered negligible.

[6] Chosen as a result of several meetings with TRAC-Monterey representatives.

unit's pre-assigned value times the probability of killing it, which would be a function of the range between the two units. The value function would then be defined as:

VPA = RedValue(s) * Pk(r)

Where:

r = Range in Kilometers

s = Scenario type

Pk(r) = Probability of kill as a function of range

RedValue(s) = Value of the red unit as a function of the scenario

Again, this example is for illustrative purpose and is a simplified version of the VPA function used in DAFS currently.

2. Constrained Value Optimizer (CVO)

Once a value function is chosen and the subsequent VPA values generated, the values are applied as the coefficients in an overall objective function designed to optimize the total benefit of all the potential assignments. Of course, the result must satisfy a given constraint set. To continue with the example above, a potential objective function may be to maximize the sum of all potential assignments from blue to red. If that were the extent of it, the solution would be easy; make all assignments that have a positive VPA. However, as is usually the case, there are limits. In our example, suppose that each blue unit may be assigned to at most one red unit and that a VPA greater than 25 is desired in each case. The subsequent formulation is then:

Maximize: $$\sum_{b \in B, r \in R} VPA_{b,r} SEL_{b,r}$$

Subject to: $$VPA_{b,r} SEL_{b,r} > 25 \qquad \forall (b \in B, r \in R)$$

$$\sum_{r \in R} SEL_{b,r} \leq 1 \qquad \forall (b \in B)$$

where: $SEL_{b,r} = \{0,1\}$

B = Set of all blue units

R = Set of all red units

Once applied, an optimized value for the sum of all the possible combinations of assignments is generated producing an assignment set. This is a relatively standard optimization problem. However, once a blue or red unit moves, or any other factor used in either the VPA formula or the subsequent optimization is changed, the assignments may not still be optimal. Managing the subsequent re-evaluation of the optimization turns out to be a key factor in the attempt to synthesize simulation and optimization.

The portion of the simulation that evaluates the battle space information and provides a solution to the implemented objective function is the Constrained Value Optimizer (CVO). The term “constrained” in the name refers to the fact that the resulting optimal solution generated by the CVO is constrained by either the passing of time or by subsequent events that may or may not invalidate the standing solution. The CVO concept is implemented in both DAFS and DAFS-ST and is described fully in the following sections.

3. Primary Assumptions

Modeling a combat environment is inherently complex. As a matter of normal analysis, several assumptions are made in order to pare the problem down to a manageable size. DAFS is no exception and involves several simplifying assumptions in the early stages of development.

First, the operating environment is flat and free of visual obstructions. Second, all contacts are instantly identified and correlated. This means that any contact that is detected or reported can be immediately correlated if it has been detected previously. Finally, target locations are considered to be accurate at time of detection. Obviously there are other simplifying assumptions incorporated in DAFS. However, they are less significant and are discussed as their relevance may be appropriate.

B. SPREADSHEET MODEL

The first phase in the analysis of a value function for use in the DAFS simulation is a base evaluation in the DAFS spreadsheet tool (DAFS-ST). The primary purpose of this tool is to evaluate potential VPA functions for sensibility and impact. Additionally, the snapshot approach provides initial insights into the critical factors associated with success of networked fires. The term snapshot refers to the fact that DAFS-ST is used only to analyze a moment in time.

The following paragraphs describe, in some detail, the approach, logic and results associated with the DAFS-ST. A more complete description of the spreadsheet and its operation is provided in Appendix 1.

1. Development

Initially, DAFS-ST was developed to provide insight into how DAFS itself would need to be approached. However, DAFS-ST soon out performed its intended use and became a valuable asset in the development process.

The workbook is actually a collection of spreadsheets designed to represent the interface, reference data and output intended in DAFS. The individual worksheets are titled inputs, generator, locations, tables, calculations, pairings, and copies. Table 2 briefly describes the function of each and refers to screenshots contained in the following pages. A complete description of each page is found in the appendix.

2. Use of DAFS-ST

As an initial evaluation tool, the primary benefit of DAFS-ST is the visual depiction of an optimized moment in time as shown in figure 2. After several automated runs under one particular configuration, a quick scroll through saved visual displays provides an excellent sensibility check of the value and objective functions applied.

WORKSHEET	FUNCTION
Input	Takes user input for number of players, operating areas, mission, acceptability limits and coverage factors. Final pairings visual display. (Figure 3)
Generator	Iterates through and allows the user to select a desired random battle space configuration.
Locations	Reference sheet once configuration is selected. Can be used for manual position entry and re-evaluation.
Tables	Contains look-up tables for P_k, threat, and player values
Calculations	Several matrix form tables that contain pre-calculated data based on the configuration and look-ups.
Pairings	Matrix form table with value function results. Resulting Assignment matrix
Copies	Historical inputs, pairings and result snapshot

Table 2. DAFS-ST Worksheet descriptions

Once the user has entered the number of players desired by type, the battlefield configuration is generated using random positions within the parameters specified. Once the battlefield is configured, the calculations begin. The computation logic in DAFS-ST follows very closely the path described in the example previously. A sequence of preliminary values for all possible interactions between unique blue and red players is calculated based on the mission, configuration and the look-up tables, the most significant of which is the VPA table that is based on the value function. After these tables are generated, the solver routine in the spreadsheet is called using constraints applied from the input table and the calculated VPA values. The solver (CVO) generates an assignment matrix, from blue to red, which is subsequently displayed on the inputs page graphically. If desired the user can quickly save the results to the copies page.

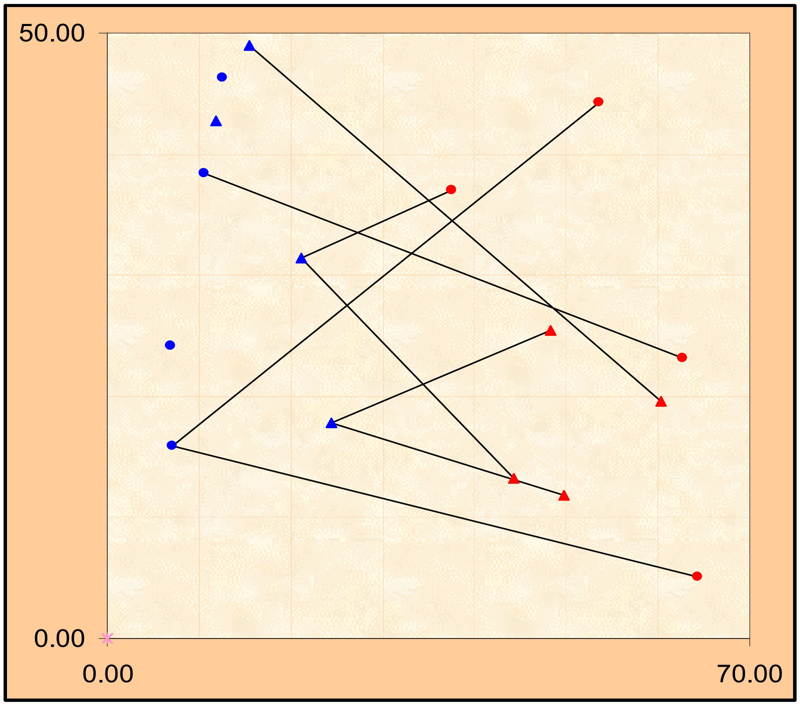


Figure 2. DAFS-ST Sample Result Display

3. Spreadsheet Benefit

Overall, DAFS-ST is very effective tool for the portion of the overall analysis that it is designed for. As an initial evaluation tool for the value function and objective function, its ease of use and simple set-up routine makes it an excellent tool in conjunction with the full model. Because DAFS-ST is used as a sensibility check, the inherent limitations of speed and variable capacities do not present any barriers to progress. The evaluations conducted using the spreadsheet may be kept to a small size without loosing any benefit in the results, allowing it to function as a fully complementing partner to the DAFS model.

C. DAFS SIMULATION APPROACH

Once DAFS-ST has been used to evaluate a potential value function and objective function, the same are implemented in DAFS. DAFS then allows the evaluation of the functions in a dynamic environment through the use of applied measures of effectiveness.

Specific DAFS structure, application and limitations are fully described in the following chapter. The following paragraphs describe the component design philosophy applied in the development of DAFS.

1. Component Structured Design

As described previously, DAFS is intended to evaluate the implications and use of networked fires over a range of potential UA design criteria and different potential networking algorithms. In order to accomplish this, the DAFS design must contain the inherent flexibility to incorporate new and yet defined configurations and implementations of FCS units and networked fires logic. To meet this need, DAFS is designed using a component philosophy for each of its functional areas.

The basis for this design philosophy is that the simulation components be designed in such a manner that they provide templates for functions and interfaces that can be implemented in a number of different ways provided the basic template is followed. This allows subsequent analysis to be conducted without extensive modification to the base simulation or its branch components. The goal is a collection of "plug-and-play" components that can be swapped out without altering the functional stability of other components. This allows the future user to define and develop advanced versions of specific components without the need to alter the base model or other components.

2. DAFS Components

DAFS' primary components are currently platforms, sensors, munitions, command elements and the CVO. Additionally, there are many sub-components that support the interoperability of these major components. For the purposes of this research the sensors, munitions, command element and CVO represent the focus of analysis. By varying the design and implementation of these specific components, the overall impact of networking fires and sensors and the success rate of different UA configurations can be evaluated. These main components comprise the foundation for analyzing whether or not networking of fires and sensors is desirable and if so, under what conditions.

Additionally, the analysis may provide insight into the best composition of an FCS force that is implementing networked fires.

III. DAFS MODEL DESCRIPTION

The Dynamic Allocation of Fires and Sensors (DAFS) simulation model extends the basic concepts demonstrated and explored initially in DAFS-ST. DAFS uses the same fundamental concepts applied in DAFS-ST and contributes the ability to analyze the same factors over time including the implementation of probabilistic events. DAFS thus gives the user a level of run-time control over the flow of the simulation not normally present in combat models. The ability to define the CVO parameters and how they affect the flow of future events as the simulation progresses allows the user to pre-assign the decision logic desired and then have the simulation adjust itself based on how events actually take place. This methodology improves on both standard simulation and optimization by applying the best characteristics of both in one decision support analysis model. This facilitates combined analysis rather than engaging in the standard practice of parallel efforts.

The following sections contain a detailed description of DAFS designed to provide the reader with an understanding of the major components of the object-oriented model, how they interact with one another, and the basic implementation steps leading to analysis. A discussion of the optimization component, the CVO, is sufficiently involved to warrant its own chapter, which follows.

A. MAJOR COMPONENTS

One of the most significant aspects of the DAFS model is its component-based architecture. Within the simulation model, there are two types of components that work together to give DAFS its overall capability. Some elements represent physical items such as sensors and munitions, others represent functionality. Both of these types of elements are equally important and allow the same benefit with respect to "plug-and-play" configuration. That is, that as individual elements, they may be switched out with different versions without altering any other portion of the model. The physical elements are platforms, sensors, weapons and munitions. The functional elements are the command element, mover managers, kill probabilities, and the CVO. In order to provide maximum understanding to the reader, the physical elements are discussed first followed by interaction fundamentals and the functional elements.

1. Physical Components

The physical components of DAFS are designed to represent actual physical components. Keeping the component interchangeability concept in the forefront, these components are designed to allow seamless replacement without affecting any other component in the simulation. As described below, the platform is the base component for a UA to which zero or more of each of the other components may be attached.

a. Platforms

Within the simulation, platforms represent the foundation structure of any Unit of Action (UA) involved in the scenario such as tanks, jeeps or unmanned arial vehicles (UAV). Platforms may also represent non-mobile entities like radar stations, but the platform element is still used as the primary reference point for all other physical elements. Platforms are only responsible for knowing their current position and velocity and reporting the same to requesting sources. Additionally, when either the magnitude or the direction of a platform's velocity vector is changed in any way, a property change is fired that may be received by other entities listening for the change. The use of listeners is key to this particular style of modeling and occurs frequently throughout DAFS[7]. The listener feature refers to the fact that an element may be programmed to listen for specific actions that occur within the simulation. These actions may be property changes or simulation events and may be triggered by other entities or by the simulation routine itself. These actions then may elicit a response on the part of the registered listener. Property change sources and listeners are resident in JAVA and the simulation event counterparts are in Simkit.

As the foundation structure for all physical entities in the simulation, the platform may have associated with it any number of the sensors, weapons and communications elements described below. One may look at this as a direct analogy to

[7] Buss (2000), Buss & Sanchez (2002)

constructing an actual combat unit where the body is first manufactured and then all of the weapons systems are installed. From an operational point of view, this means that wherever the foundation, or body, of the unit goes, so does the attached system. Therefore, the only entity that really needs to know its location, is the foundation, or the platform. In the case of DAFS, once the platform entity is created, the associated sensors, weapons, munitions and communications are given a reference to the platform as they are created. This is conceptually depicted in Figure 3.

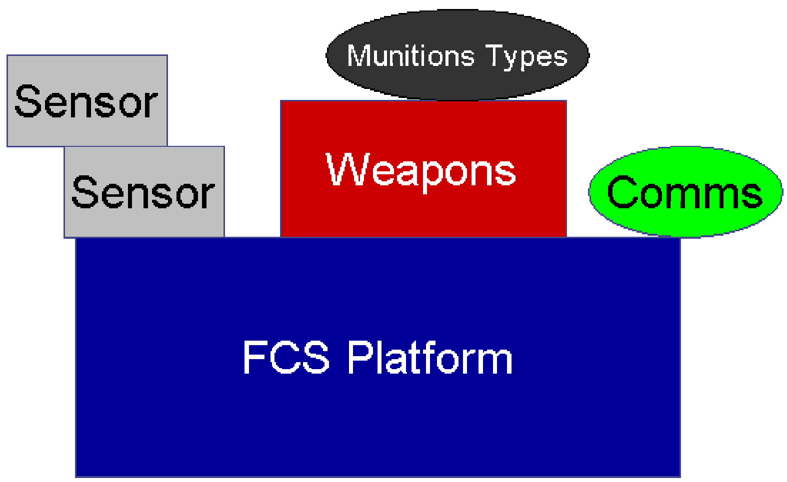


Figure 3. Entity Structure Example

b. Sensors

Like the platform, the sensor component is very limited in its functionality. The sensor maintains only its basic capabilities in the form of its type, max range and footprint. Additionally, it maintains a container for its detections. Once created, a sensor object is given a reference to its associated platform in order to locate itself. The capability of a sensor to process detections is accomplished through the use of the functional objects called mediators and referees along with the listening process described earlier. The concept of referees and mediators, or adjudicators, is repeated between munitions and targets and a description of how all these items interact is contained in the primary interactions section below.

c. *Weapons*

From a functional point of view, the only task that a weapon accomplishes is launching munitions. Thus, the weapon itself does not play a critical role in the basic analysis of the effectiveness of the force. It is the munitions that interact with targets and therefore represent the real objects of focus when it comes to combat adjudication. The weapon object has been developed though to allow more analysis of UA configurations. Specifically, because a weapon object defines a platforms ability to deliver particular munitions types, the collection of weapons configured into a platform largely defines its potential employment.

d. *Munitions*

Munitions objects draw on the same process used to make the sensors function. Like the sensor, the munitions object only keeps track of its type and footprint. The adjudication of a weapon-target interaction is handled by the referee/adjudicator combination described below. At runtime, only the inventories of munitions by type are established on each platform. During the running of a simulation, if a munitions object is needed, it is instantiated on the fly provided the inventory level is greater than zero. This methodology minimizes the number of active objects in the simulation and improves performance.

2. Primary Component Interactions

There are two primary interaction templates that give DAFS the majority of its functional capability. The first is the referee-mediator and referee-adjudicator template, which apply to sensors and munitions respectively. The second is the source-listener template that allows two things. First, it allows the monitoring property changes throughout the model as a data gathering medium for analysis and second, as briefly described above, it allows elements within the simulation to act based on the actions or property changes of other elements. Table 3 captures the basic organization and function of these templates.

Template	Type	Function
Referee/Mediator	Referee/Sensor Mediator	Determines platform interactions
	Referee/Munitions Adjudicator	Determines munitions effects
Source/Listener	Property Change	Triggers actions based on the state change of another entity
	Simulation Event	Triggers actions based on the occurrence of a particular event

Table 3. Interaction Templates

a. Sources and Listeners

The source-listener protocol is essential to the success of discrete event programming. As a tool, the protocol is one of the items that separates discrete event simulation from time step simulation. In time step simulation, all potential interactions must be checked at each time step to resolve whether or not an interaction is occurring, an evaluation load on the order of N^2 for each time step, where N represents the total number of entities in the scenario. This also means that interactions that would have begun in the mid-point of the time step are delayed and thus alter the level of "reality" attained. Discrete event simulation, on the other hand, by implementing the source-listener template, calculates the precise time of interactions and schedules the event at that time. At most this requires an evaluation load on the order of N for every event or property change. As the events are reached on the event list, the appropriate actions are taken, and the simulation continues.

The two main uses for the source-listener template are simulation control and data gathering. However, both function in exactly the same manner. The primary difference being that when a data gathering listener "hears" a change in the simulation, it only records the information and does not subsequently affect the remainder of the simulation run.

The key elements of the source-listener template are the sources, the listeners and the registration process between the two. Sources, as the name implies, are the source of a trigger that may or may not require action on the part of another entity within the simulation. It doesn't matter if there is a registered listener or not, if it is something that could affect something else, the source is responsible to "fire" the information. The listener is the receiver of this information, and is responsible to process it however it has been programmed to do so.

The critical link is the registration process. The listener must be registered as such with the source in order to receive the information. This registration process provides the benefit of reducing the processing load to only those entities that have the need or capability to deal with the particular information fired. For example, a detections counter would be registered as a listener to a particular sensor, the source. Every time the sensor fires a detection event, the counter will hear it and tally that a detection event had occurred. This is an example of a data-gathering listener. If the parent platform of the sensor was also registered as a listener, it may alter its course as a result of the detection event. That would be a simulation control item.

Sources and listeners are used extensively throughout DAFS. One of the most impacting uses is in the evaluation of interactions between weapons or sensors and the platforms in the battle space. For this application, referees are registered as listeners and oversee the potential for interactions.

b. Referees and Mediators/Adjudicators

The referee-mediator/adjudicator template is used extensively in DAFS. The concept of mediators and adjudicators is exactly the same except that the mediator applies to sensor-target interactions and the adjudicator to munitions-target interactions. For the sake of brevity, only the referee-mediator template is described here with references to the adjudicators as necessary.

The referee may be viewed as a simulation monitor that listen for changes within the simulation that may lead to interactions between entities, or to changes in previously determined interactions. These events could be the appearance of a new entity, a change in an entity's velocity vector or the detectable properties that an entity may be emitting. In essence, a referee is a focused "eye-in-the-sky" that monitors whether changes in entities it is responsible for might result in subsequent interactions. Once this potential is determined, the referee passes entities that may have interactions to the appropriate mediator or adjudicator.

In the case of sensors and targets, the referee listens for changes in targets that would potentially create or change a detection event. If, for instance, a target maneuvers, the referee hears the change and executes its process. The referee takes the target's new course and speed, and with it, determines what sensors the target will come within range of. The referee only considers the sensors that have the ability to detect the target. For each of the sensors that will have the target enter its footprint, the referee passes the target and the sensor information to a mediator. The mediator then uses the detection algorithm associated with its footprint to determine whether or not a detection event will occur. If so, the detection will be scheduled on the event list and the simulation will go on. If not, nothing occurs. If the sensor already has a detection scheduled for a particular target and it will no longer occur, or will be different, the appropriate changes are made.

The referee-adjudicator template follows the same logic described for the referee-mediator and is applied when a munitions object fires an impact event. The referee then accomplishes the same task with the munitions footprint and the targets within it. Adjudicators determine the extent of damage occurring to targets based on the munitions type and distance from the impact.

3. Functional Components

Functional components within a simulation handle administrative matters and serve as decision or organization modules. Within DAFS, there are three significant functional components that will be discussed: mover managers, command elements and kill probability objects. Additionally, the inventory object will be described. The inventory object does not, at this point, play a critical role in the simulation. However, its concept and functionality will become increasingly beneficial as the research in this area grows more complex.

a. Mover Managers

Mover managers, as the name implies, manage the movement behaviors of the platforms. Each mover manager object type represents a specific movement pattern that a platform may engage in. Current forms of mover managers are patrolling, intercepting and basic path following. Each mover manager gets its unique form through different combinations of location control and behavior. Each uses JAVA Point2D objects for location management and simulation event protocol for its behavior. Table 4 summarizes these mover manager types.

b. Command Element

The command element is a functional element associated with each platform. This element organizes priorities, objectives and capabilities within each unit. The command element has two primary functions. First, it acts as a priority filter to keep the highest desired action at the top of the list. Second, it maintains track over requirements, such as reporting criteria or munitions inventory status, and ensures the entity complies with actions as necessary. The command element makes use of the listener protocol to accomplish its monitoring functions. It is the command control element that controls which of the mover managers is currently being used by the platform and whether or not it will engage targets within range.

Mover Manager	Location control	Behavior
PathMoverManager	List of JAVA Point2D objects	Sequences through the list of points and stops at the end.
PatrolMoverManager	List of JAVA Point2D objects	Sequences through the list of points and repeats a set number of times or unlimited until another mover manager takes control.
InterceptMoverManager	Single JAVA Point2D object	Proceeds to the point and triggers the behavior contained.

Table 4. Mover Manager Descriptions

c. Kill Probability Objects

Kill probability objects contain the ability to generate the expected probability of kill for a particular munitions type against a particular platform type as a function of range. The basic template for these objects does not presume the method that will be used to generate the value. Rather, the kill probability interface requires a contract set of methods that the user must employ so that any kill probability generator will work. Kill probability implementations currently in DAFS include linear, piecewise linear and exponential functions. A kill probability implementation that utilizes a lookup table was also developed. Other functional forms may be developed and used, as long as the kill probability interface is implemented.

d. Inventory Objects

Also stemming from an interface, inventory objects were developed to allow DAFS some level of benefit from logistic considerations. The interface for this object defines basic inventory methods including adding inventory, reducing inventory, returning the level for a specific item and many other standard inventory functions. Currently, the inventory object is used to track munitions inventory levels to assist in both the VPA calculation and eventual use of munitions. Again, because the objects stem from an interface, the user may design several other inventory objects for specific purposes and give them additional methods required to complete the functionality desired.

B. IMPLEMENTATION

The implementation of DAFS can be broken down into three distinct areas: input, runtime and output. DAFS input is a collection of XML files that predefine every aspect of the participants, the scenario, the nature of the runs and the desired output. Many of the input components are independent of one another and therefore may be altered or replaced without any affect on the remaining pieces. Others contain several related components and must therefore be altered as a whole. However, sub elements within these larger input files may be swapped out in the same manner as long as the integrity of the overall file remains. Runtime for DAFS is consists of a standard discrete event simulation run that contains entities that are intermittently controlled by the use of a local optimization routine. The output is available in a number of formats and again, is dictated by input XML files. The user has the choice of displaying output to the screen, writing to files, generating XML files or any combination. XML output files are particularly beneficial as they may be altered using XML stylesheets or queried in a number of ways to present the results.

1. Input

Figure 4 on the following page is a graphic representation of the input scheme used by DAFS. Each of the blocks on the left side of the diagram represent a self contained XML document and the significant contents. From this diagram, it can be seen that the simulation entities input file must contain a significant amount of information, which is due to the nature of constructing a UA for participation. Because the components used in the construction of a unique platform are closely tied to each other, with respect to references, they must be generated at the same time so that the proper associations can be made. This does not mean that every entity must contain each of the listed items; it simply means that if any of the listed items are going to be a part of the entity, it must be contained in the appropriate XML tag structure associated with construction of an entity. The remaining blocks on the left side of the figure also represent potentially discrete input files, each having a particular tag structure.

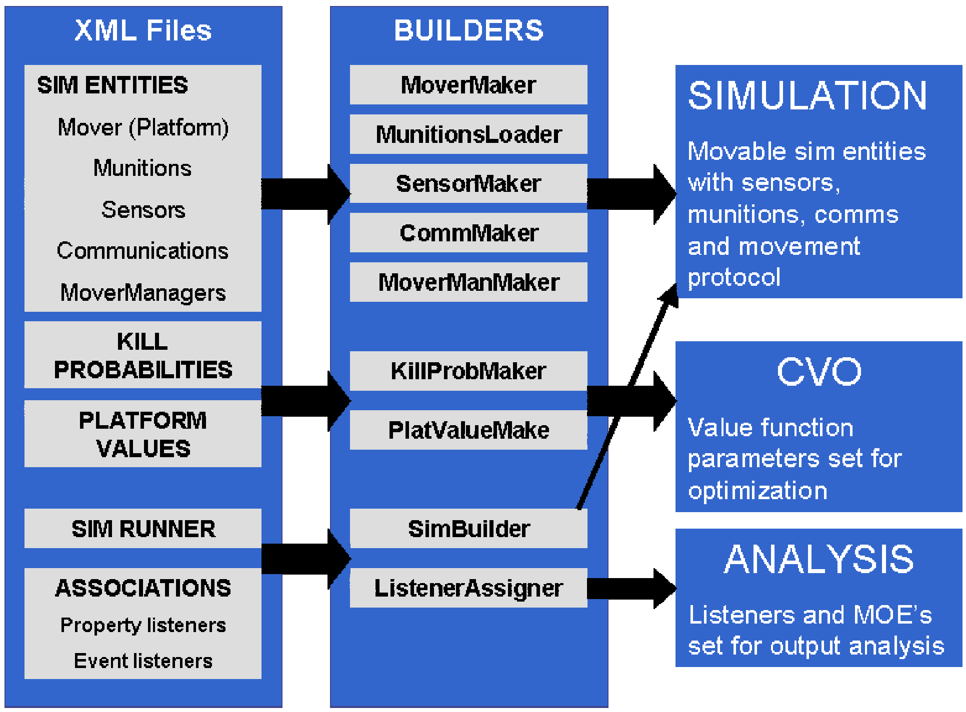


Figure 4. DAFS Input Design Structure

As the input files are discussed in the following paragraphs, the reader may find it useful to refer to Appendix 2, which contains sample XML files. The experienced XML user will note that the structures of the input files may be defined and validated through the use of Document Type Definitions (DTD) or SCHEMA documents however.

The kill probabilities file contains the necessary information to generate kill probability object instances discussed in the previous section. Each kill probability instance covers all engagements between a particular munitions type and a particular platform type. Therefore, once the user has defined all of the possible interactions between munitions and platforms, this file does not need to be modified. When a new munitions type of platform type is desired in the scenario, the user may define new kill probabilities for the new entity and add them to the existing file. Kill probability instances should be generated to take into account friendly fire issues. Recall, the munitions-target referee will evaluate all targets within the munitions footprint regardless of the affiliation.

The platform values file is another file that once generated, can be used for all runs. Within the file, each platform type is assigned a value for a given scenario type. Once the user is happy with the choices, the file does not need to be altered unless new platforms are added or a different scenario type has been developed. Conversely, this file represents an excellent choice of a design point for analysis.

This simulation runner file contains the information necessary to implement the Schedule class in Simkit. This file contains the parameters that define the simulation stop criteria, non-data output options and the number of repetitions desired. The stopping criteria may either be set to an elapsed time or to the occurrence of a specific event, the tenth kill for example. The non-data output options refer to the simulation event output. The two categories are verbose and single step. If verbose is set to true, the simulation will generate an event list to the screen at each event change while the simulation is running. A selection of false will yield nothing. Single step will control the simulation by allowing it to progress one event at a time and, by definition, will invoke the verbose output method. This allows the user to view each discrete event as it occurs. The repetitions selection will reset all components to the original configuration and begin the simulation again. This is particularly beneficial for multiple run analysis of probabilistic scenarios as the simulation will begin the same but will not provide the same exact run due to the implementation of different random numbers.

The associations XML file is used to assign the listeners not already prescribed by DAFS. DAFS automatically registers the appropriate listeners necessary to accomplish successful running of the simulation. The associations contained in this file are for data gathering purposes.

The middle block of Figure 4 represents the collection of XML builders that take portions of the XML documents and use them to create the necessary JAVA objects. The DAFS main method is responsible for farming out the appropriate XML elements to the correct builder. In all cases, with the exception of the munitions loader and the listener assigner, the XML builder class instantiates JAVA objects corresponding with the name of the class. For instance, the mover maker instantiates a Simkit Mover object, the base component for each UA in the simulation. These were presented earlier as platforms.

The listener assigner and the munitions loader each have slightly different functions but do still convert XML file data into simulation information. The munitions loader by munitions type, loads the inventory object on each platform with the corresponding number of rounds as initial inventory. Again referring to Figure 4, the munitions information comes from within the large XML file of simulation entities. Specifically, the munitions element is a sub element of the mover element. This structure is what allows the loader to associate the munitions with the correct platform. An example of a simulation entities file is contained in Appendix B.

The listener assigner is primarily to establish data gathering connections for simulation monitors. Typically simple statistics objects, these monitors listen to the objects to which they are assigned or to the simulation in general and tabulate events or property changes. The builder file in this case serves to register the appropriate objects as listeners. These monitor objects and their configuration is essential to retrieving usable output from a simulation run and the concepts are discussed more fully in the output section below.

2. Runtime

A DAFS simulation scenario currently involves two sides, red and blue, although there could be an arbitrary number of sides. Each side is given its objectives through the implementation of the mover managers and the level of aggressiveness protocol assigned to the command element. The mover managers dictate where and how the platforms will proceed as the simulation progresses and the aggressiveness factors dictate how the platform will behave upon interaction with other platforms.

Additionally, the blue side is provided a scenario posture, which affects the player values on both sides and has subsequent impact on the VPA values as they are calculated. The implementation of the CVO is only accomplished for the blue side and assumes the red side is using less sophisticated operational capabilities. Namely, the red side is assumed to operate as a conventional force with standing orders for objectives and rules of engagement (ROE) set from the beginning. The point of DAFS, at least initially, is to explore whether or not the networking of fires and sensors by a force has greater effectiveness than fighting with pre-designated routes, assignments, and ROE. Therefore, initial analysis with DAFS does not assume that the opponent is implementing the same technology so there is a visible difference in the results if indeed the networking effort has an affect.

The command control object associated with each platform provides it with a unique engagement behavior. When a platform of one side detects an opponent platform, as in the real world, it must do some analysis as to its course of action to follow. In the case of DAFS platforms, this is accomplished through its ROE in the command object to determine whether or not to engage. If the platform determines not to engage, the command element will dictate in what manner the platform will avoid engagement and implement the appropriate mover manager. This once again highlights the component nature of the DAFS simulation and its resident flexibility. Rather than employ a single mover manager with differing methods for the particular behaviors, each mover manager is a distinct object that can be removed, replaced or added. This allows the user to maintain behaviors that have proven successful and change only those that need further development.

If the platform elects to engage, the engagement protocol for the particular munitions will be called. This may implement a delay time designed to emulate set-up times associated with particular delivery systems. Currently this emulation is based purely on the munitions type and does not account for different delivery systems for the same munitions type.

The simulation will run in this manner until the designated stopping a criterion has been met. Upon completion, the output that was designated during the XML input process will be gathered and output according to the selected output methodology.

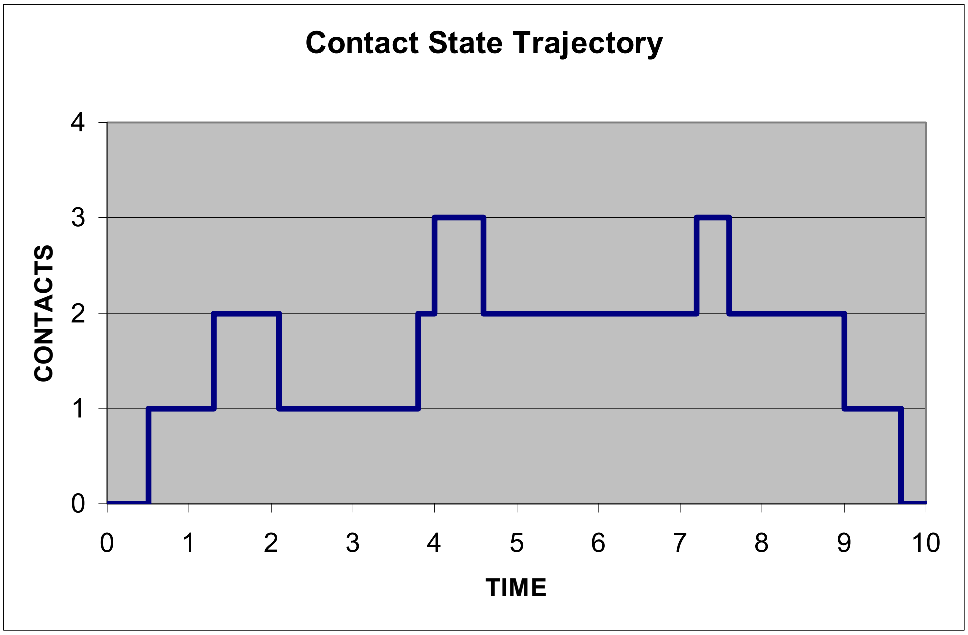


Figure 5. Example State Trajectory

3. Output

Through extensive use of the listener functions described earlier, statistical objects are created and tasked with monitoring specific events within the simulation. As a result, the desired output is "programmed in" to any specific simulation run and subsequently provided to the user in a predetermined format. The tally and time varying statistics objects used for data gathering are both resident in Simkit. The tally version keeps track of simple values that only require counting, such as the number of red players killed or the number of missiles used. The time varying version keeps track of the level of a particular state and the corresponding times when the value changes. This state trajectory can then be used to retrieve quantitative values with respect to time, such as the number average number of contacts held or total time with a certain number of contacts held. Figure 5. demonstrates an example state trajectory for the number of contact held by a particular platform. Because the information is retained with time information, the time varying statistics object is capable of returning values as a function of time. In this case, the time-averaged mean of contacts would be computed by dividing the area under the curve by the current time.

Through the use of XML file writing functions in JDOM, the output values collected by the statistics objects can be selectively written to output XML files for future analysis. As an option, the output information may be written to the screen or to output text files. The various outputs are selected at runtime through the input process and the associations input document. XML output files are extremely beneficial to the user because they can be manipulated in a number of ways to present the output. Through the use of XML stylesheets, or XSL documents, the output values can be selectively extracted and displayed in several forms including web pages and as graphs.

IV. CONSTRAINED VALUE OPTIMIZER (CVO)

The Constrained Value Optimizer (CVO) is the entity in the simulation through which decision factors are used to generate a local optimal solution. When applied, the CVO solution enables the forces in the simulation to revise their collective engagement tactics to increase the near term probability of success. The goal, through successive implementation, is to develop an on the fly scripted tactical approach to the unique situation.

A proposal to implement global optimization techniques in combat analysis would be a lofty venture indeed. At best, global optimal solutions take stochastic events into consideration only as they may be estimated, that is over several successive instances. Conversely, a near global solution for a *particular* situation may be achieved by the successive implementation of local optima based on an evaluation of the environment as stochastic events occur; a piece-wise, near optimal solution.

The two main components involved in this process are the CVO and its associated Value of Potential Assignment (VPA) object. The CVO holds a linear programming (LP) formulation, calls an LP solver package and handles the returned solution. The VPA, as it was in DAFS-ST, is a pre-processing tool that populates the objective function coefficients in the CVO LP. Together, the CVO and the VPA work to allow DAFS the ability to make and invoke logically derived decisions with respect asset allocations.

A. CVO COMPONENTS DESCRIPTION

As with the simulation configuration and simulation control components of DAFS, the CVO is part of a component-based design that allows flexibility in configuration and analysis. The CVO and VPA are both interfaces that allow multiple implementation versions. For ease of discussion, the CVO has, to this point, been referred to without reference to the VPA. This is because the CVO actually *contains* a reference to an implementation of a VPA that it calls as necessary. When DAFS is running, a direct call, a time interval, an event counter, or some combination, triggers the CVO local optimization routine. The routine employs the CVO, VPA and the LP_Solve software package. A representation of the functional relationship between the CVO, VPA and LP_Solve is presented in Figure 6. The sequence of events is started by the trigger, which causes the CVO to request objective function values from the VPA (1). The VPA returns the coefficients (2), which triggers the solve call (3). LP_Solve returns the solved formulation or an indication of no optimal solution (4).

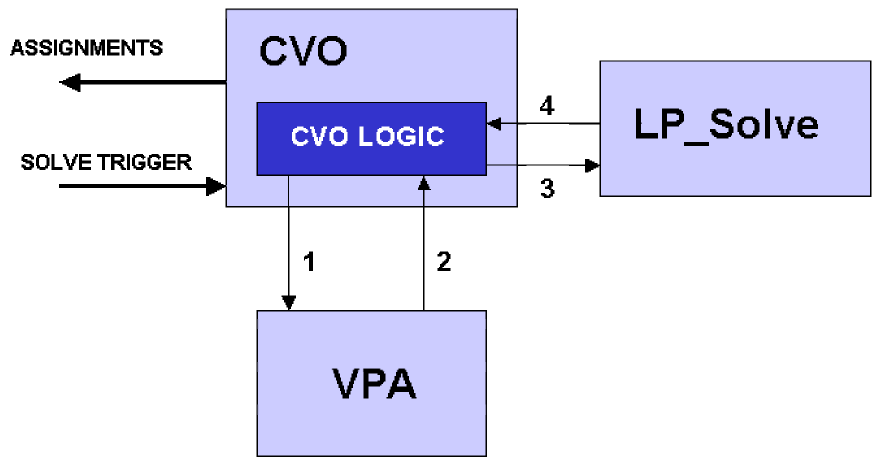


Figure 6. CVO Logic Flow

1. LP Software

The objective function formulated in the CVO is solved using a JAVA version of LP_Solve 2.0, which is freely available for download. LP_Solve is a package of LP solver methods that implements the simplex method. LP_Solve version 2.0 is the latest version to have a JAVA implementation and therefore the necessary version for use with DAFS.

LP_Solve is reported by its author to be capable of handling up to 30,000 variables and up to 50,000 constraints; a sufficiently large numbers for DAFS implementation. In the current formulation developed in the CVO, the number of variables is the product of the number of platforms on each side, and the number of constraints is the product of the number of blue platforms and two times the number of red. A configuration of DAFS that has, at any moment, 100 units per side, is well within the stated limits of LP_Solve. This is sufficiently large to explore the factors described previously.

To confirm the validity of LP_Solve solutions and the implementation, solutions generated were compared to the DAFS-ST solutions and in all cases were exactly the same or slightly better. Because of the tolerance levels in the resident solver package in Excel, LP_Solve was sometimes able to find a slightly better solution than Excel.

2. CVO

In DAFS, CVO is defined as an interface and several classes implement CVO. Each instance of CVO serves as an interaction module for communications between LP_Solve and DAFS. DAFS is currently capable of containing multiple CVO objects that can be implemented to allocate different assets. For example, two CVO objects may be employed by DAFS and both fires and sensors may be allocated. A third may be added and the allocation of re-supply assets considered. There is no theoretical limit to the number of assets that could be allocated through multiple implementations of CVO objects. Because each CVO object formulates its LP differently, the user may employ the same logic to different assets, or a different one to each. The formulations associated with the current implementations of CVO objects are explained in section B to follow. As will be shown, the formulations implemented by the CVO objects are very straightforward. The formulations used for the VPA objects are more complex and are representative of the breadth of factors that may be employed in a decision cycle.

3. VPA

The VPA is another component in DAFS that is an interface for which several implementations have been developed and tested. On the whole, they represent heuristics currently used by the Army in the allocation of assets. The formulations include the consideration of munitions types, position, value of the target and many other factors. Due to the interface, multiple implementations of VPA objects are possible and may be used in a DAFS simulation with no impact on the CVO or its formulation.

The function of a VPA is very simple and follows the template developed in DAFS-ST. It takes in sets of blue and red entities, uses them to evaluate the VPA value for each potential pairing, and returns a list of VPA values, one for each pairing. These values are then applied as the objective function coefficients in the CVO formulation. The experienced JAVA user should note that the reference to sets and lists in the previous explanation does not imply the use of the JAVA classes by the same name.

B. FORMULATION

As in DAFS-ST, optimizing the assignments in the battle space is a two-step process. When triggered, the CVO first directs the VPA to generate values that are then passed to the CVO for use as the objective function coefficients in the optimization. The second step is solving the resulting optimization model. The discussion below will present representative formulations for both the VPA and the CVO. The formulations discussed were chosen because they are the versions used in the analysis presented in the following chapter. However, the reader is reminded that the CVO and the VPA are both functional components that may be replaced by implementations designed differently.

1. VPA Formulation

Aside from being the means to assign values to potential assignments within the battle space, the VPA also serves as a means to pre-screen acceptable values and thus eliminates the need for additional constraints in the optimization. Though LP_Solve has not been challenged by the scope of problems implemented to-date, reducing the number of constraints allows more room for scaling up and reduces the complexity involved in defining the optimization problem.

Several of the VPA formulations that have been implemented in DAFS are defined below and explained. Where appropriate, the formulations used for analysis discussed in the following chapter are indicated by an asterisk.

The variable set used in the formulations consists of all data variables and is defined as:

X = Set of tactical values for red units, defined by type and blue mission.

$x \in X$

Y = Set of tactical value for blue units, defined by type and mission.

$y \in Y$

p = Best range-based P_k value from the possible set of munitions.

Note: "Best" depends on the formulation.

q = Best predicted range-based P_k the red unit can employ.

s = Binary factor based on BestPK acceptability.

t = Binary factor based on ThreatPK acceptability.

r = range between red and blue units.

δ = Percentage penalty associated with urgency of need to engage.

c = 1 or 0; Designate capability required.

d = 1 or 0; Designate capable unit.

σ = 1 (in all cases, 1 is subtracted to prevent the potential consideration of a zero coefficient during the subsequent maximization in the CVO. This reduces the potential for multiple optimal solutions)

<u>Fires Allocation; movement to engage not considered</u>

$$VPA = \{[x \times p - y \times q] \times s \times t\} - \sigma \qquad (1)$$

p = Best P_k found considering all munitions at the current range.

<u>Fires Allocation; movement to engage considered*</u>

Formulation (1) with the following: (2)

p = Best P_k found, considering the smallest of the current range or the maximum effective range, for each munitions.

Fires Allocation; movement to engage considered and penalized

$$VPA = \{[x \times p - y \times q] \times s \times t\} - \delta - \sigma \tag{3}$$

p = Best P_k found, considering the smallest of the current range or the maximum effective range, for each munitions.

Sensor/BDA Allocation; Range dependant*

$$VPA = \{A \times [(1/R) + 1000]\} - \sigma \tag{4}$$

where:

$$A = \begin{Bmatrix} 1 & c \geq d \\ 0 & o.w. \end{Bmatrix}$$

Designate assignments are considered an allocation of sensors.

Formulations (1), (2) and (3) are cost-benefit evaluations associated with a potential fires assignment. In each case, the term inside the square brackets represents the expected gain minus the expected loss. From these two formulations it is clear that one of the critical factors of initial analysis is the tactical value assigned to the units based on mission type.

The difference between formulations (1) and (2) is that in (1) an out of range contact is not given any potential for assignment. In (2) and (3), movement to engage is considered with (3) including a penalty for the delay in engaging. This takes into account the urgency of need to engage targets based on contact density or target value.

Formulation (4) has three potential applications. It may be used to evaluate assigning sensors to an area to; one, conduct reconnaissance, two, conduct BDA or three, to designate or spot a target for Beyond Line of Sight (BLOS) fires. The expression within the brackets inverts the range, so a shorter range is more favorable, and scales the value. The variable *A* represents a Boolean condition that will be zero if designate is desired but the unit is not capable, which will drive the VPA value to negative one.

2. CVO Formulation

Once the VPA values have been calculated they are transferred to the CVO to be incorporated as the objective function coefficients. All implementations of the CVO in DAFS to date have utilized the same formulation presented here. The formulation is a multi-dimensional knapsack problem with set covering and generalized set packing constraints. The set packing constraints are considered generalized as they do not follow the convention of setting the value on the right hand side to one. The formulation presented here has been implemented in both fires and sensors Allocation CVO objects.

Sets and Indices

I,	Set of blue platforms, $i \in I$
J,	Set of red platforms, $j \in J$

Data

MaxAssign,	constraint on maximum red unit assignments that may be given to a blue unit.
MaxCover,	constraint on number of blue units that may be assigned to a particular red unit.
MinCover,	minimum number of blue units required for assignment to each red unit.
$VPA_{i,j}$,	Value of Assignment from blue unit i to red unit j.

Decision Variable

$X_{i,j}$,	1 if blue unit i is assigned red unit j.

Formulation

Maximize

$$\sum_{i\in I, j\in J} C_{i,j} X_{i,j} \tag{5}$$

Subject To:

$$\sum_{j\in J} X_{i,j} \le MaxAssign \qquad \forall (i \in I) \tag{6}$$

$$\sum_{i\in I} X_{i,j} \le MaxCover \qquad \forall (j \in J) \tag{7}$$

$$\sum_{i\in I} X_{i,j} \ge MinCover \qquad \forall (j \in J) \tag{8}$$

$$X_{i,j} \in \{0,1\} \qquad \forall (i \in I, j \in J) \tag{9}$$

Description of the Problem

The objective function (5) maximizes the sum of the values based on selected assignments. The generalized set packing constraint (6) requires that each blue unit be assigned no more red units than the value of MaxAssign. Likewise, the generalized set packing constraint (7) requires that no more than the value of MaxCover blue units be assigned to any particular red unit. The set covering constraint (8) requires at least the value of MinCover blue units be assigned to each red unit. MinCover is normally set at either zero or one as multiple units will be assigned provided (6) and (7) are not violated and the objective function value can be increased as a result. Finally, constraint (9) dictates that the decision variable $X_{i,j}$ be binary.

C. SIMULATION INTERFACE

The extent of interface channels between DAFS and the CVO is minimal. As the simulation runs, the CVO is notified of participating units on both sides. Initially, all blue side members are registered with the CVO and the red units are added as they are discovered. In baseline analysis models, the red units are also registered initially to simulate a developed battle space. The classification of red units is assumed at detection, enabling the CVO access to the necessary information for evaluation.

After each optimization, the CVO transmits the assignments determined to each blue unit. This is a direct communication and each unit in the battle space is not aware of, nor does it take into consideration, the assignments of other units. The assignments are received by the command element and prioritized. If a blue unit is engaged at the time of receipt of assignments, the engagement is completed prior to action based on the new assignments.

THIS PAGE INTENTIONALLY LEFT BLANK

V. ANALYSIS USING DAFS

With specific guidance from TRAC-Monterey and using current projections for Future Combat System (FCS) entity descriptions and considerations, a scenario was developed to conduct factor analysis. The scenario is a standard Army tactical scenario involving a perceived battalion level Unit of Action (UA) versus elements of a red brigade. The objective for both forces in the scenario is the securing of a region in the center of a battle space designed to emulate a strong point such as a town or airfield. The base scenario description provided below represents the full scale model and was reduced for analysis production runs.

A. BASE SCENARIO DESCRIPTION

The initial scenario represents a battalion sized Unit of Action employed in the timeframe of 2014. This fits into the timeframe and vignettes proposed in The US Army's TRADOC Pamphlet 525-3-90/O&O, Operation and Organization Plan for the Maneuver Unit of Action.

The battalion size UA was chosen for the experiment because it is the lowest level unit that could potentially benefit from networked fires. At this level a variety of munitions types could be examined including the long-range deep strike weapons such as the Non Line of Sight Launch System (Specifically a Precision Strike Munition (PAM)) and future mortar systems. Also stated in TRADOC Pam 525-3-90, is that the Battalion UA is the principle maneuver unit capable of independent operations. Therefore it can be assumed that it would have appropriate elements task organized from its higher headquarters, to include a robust Unmanned Arial Vehicle (UAV) sortie. The Composition and Munitions considered are represented in Figure 7.

Movement to Contact was chosen as the mission for the analysis set. This mission type best allows all the aspects built into DAFS to be shown and represents a mission where the operating picture would likely be developed on the run. Through this mission the mover managers, the detection and engagement adjudication processes, and the execution of battle damage assessment (BDA) are all exercised.

Both the UA and opposing forces have the same mission and therefore the entire action is best described as a meeting engagement where both forces have the same objective area; a 2km box in the middle of the 100km^2 battle space.

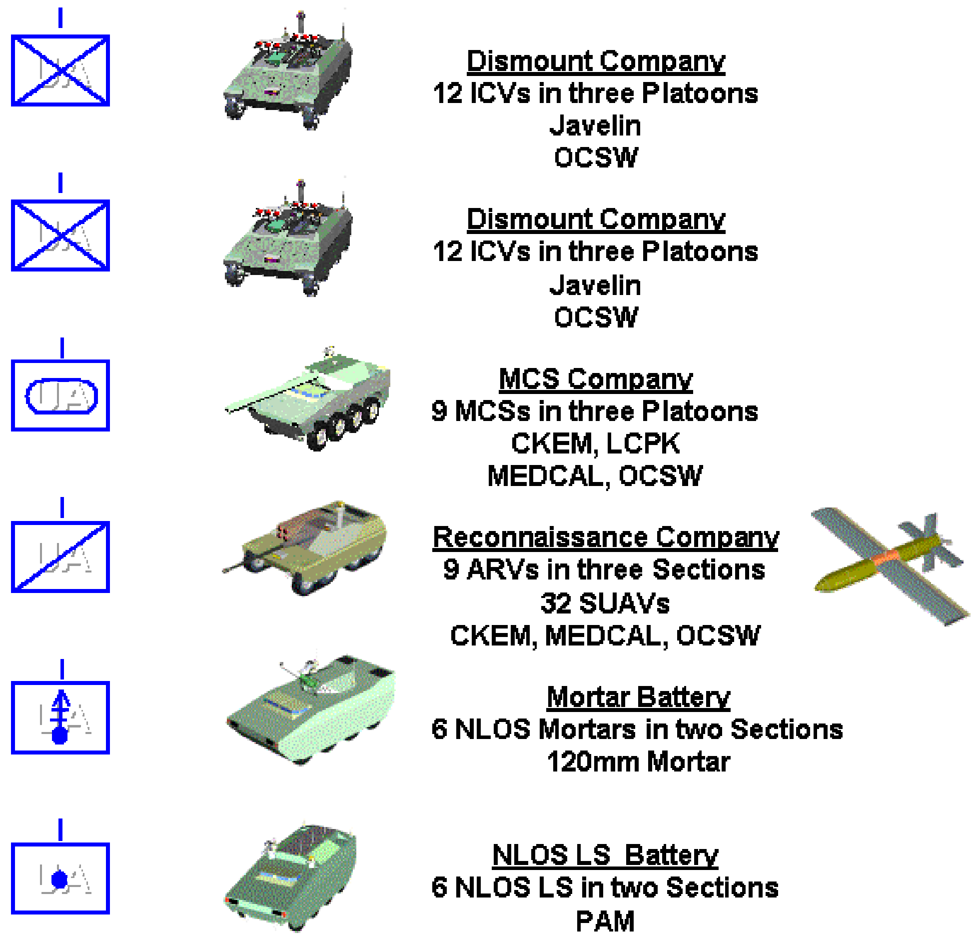


Figure 7. Blue Force Composition

The opposing force (OPFOR) in the scenario represents a world-class force that could be met in the time range of 2014. The OPFOR is designed to have a near two to one advantage on the UA and to be an armor heavy force. This is in line with the Army's hypothesis that a UA Force has the ability to defeat a force three times its strength. However, the initial analysis conducted is focused on factors contributing to success in networked fires and is impacted little by the ratio. As stated, the OPFOR has the same movement to contact mission to the same objective, and start with a small reconnaissance force within the Objective area. The OPFOR's limitations include not having networked fires, nor having Beyond Line of Sight (BLOS) munitions. The compositions of the OPFOR is represented in Figure 8.

The scenario begins with an OPFOR force lightly occupying OBJECTIVE ALPHA with company reconnaissance element. Four battalion (-) elements occupy four separate assembly areas and are on the verge of crossing lines of departures. The following narrative lists their goals as initially programmed. Their orders generally state that the tank units will move forward and assault, then occupy security positions outside of OBJECTIVE ALPHA. The OPFOR Mechanized Infantry goes to defensive positions within OBJECTIVE ALPHA. Two other reconnaissance companies maintain security on the flanks of the OPFOR attack and secure objectives on either flank of the main OBJECTIVE ALPHA.

Figure 8. Red Force Composition

The mission for the UA Task Force is also to secure OBJECTIVE ALPHA. Task organized to the UA commander is a standard SUAV sortie of at least 32 SUAVs that are on station in search patterns in a surveillance area over the OPFOR line of departure. For the UA, this surveillance area is also know as ENGAGEMENT AREA BRAVO. The UA commander already has a Common Operational Picture (COP) of specific enemy locations. It is the enemy's dynamic response that will cause uncertainty for the UA commander in the progression of the fight. As programmed, the SUAVs' individual missions are to check BDA and acquire new targets (an enemy platform that escapes the first volley can become a new target).

The main attack for the UA occurs with two infantry and one tank company. In Objective Force language this equates to 24 Infantry Carrier Vehicles (ICVs) equipped primarily with Javelin and 18 Maneuver Combat Systems (MCS?) primarily equipped with a CKEM like munitions.

In support of this UA attack are a battery of Non Line of Sight Launch System (NLOS LS) and Mortars (NLOS Mortars). These operate in sections at standoff distance in firing positions to support attacks into the EA BRAVO and on OBJECTIVE ALPHA as directed by the CVO (fire solution).

Also providing security and close observation on the objective are three companies of Armed Reconnaissance Vehicles. One Company is up front and provides attack handoff at PHASE LINE (PL) CHARLIE. The other two ARV companies provide flank security on the attack and progress forward. This entire operation is represented graphically in Figure 9.

As stated, his scenario fits well within the vignettes as described in TRADOC Pamphlet 525-3-90/O&O, Operation and Organization Plan for the Maneuver Unit of Action. The munitions, platform and sensor capabilities were developed primarily with the use of the Unit of Action Systems Book, published by AMSAA (14 AUG 02) and professional military judgment. The programmed parameters include platform speed and armor hardness, munitions range (to include minimum engagement range), Probability of hit and blast effect radius.

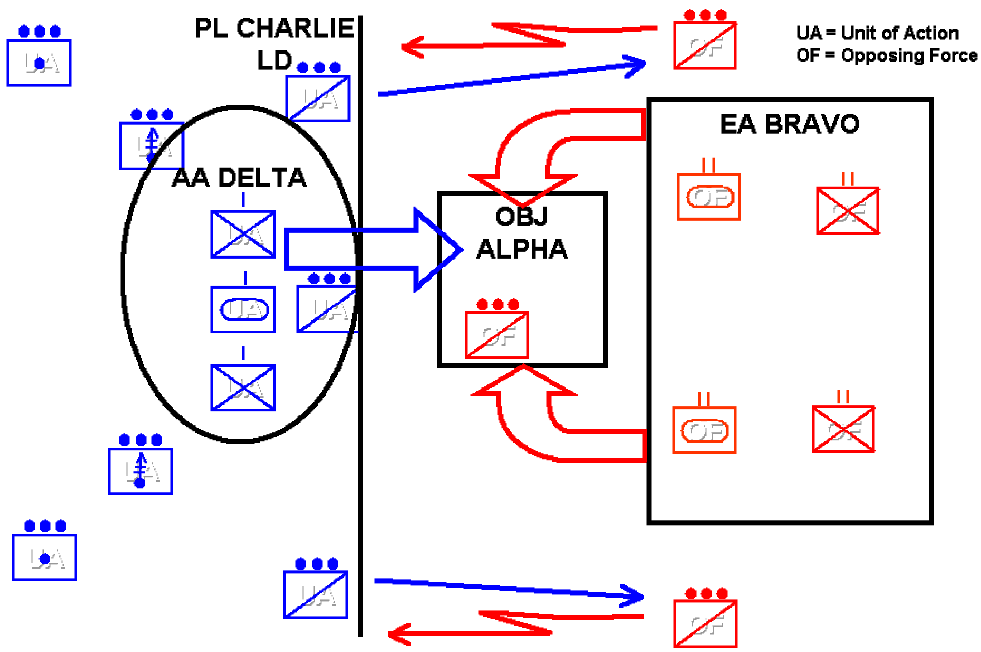


Figure 9. Full Scenario

B. DESIGN OF EXPERIMENT

In order to accomplish timely analysis, the scenario described above was reduced to size to balance model run time and output relevance. Table 5 represents the number of units used by type. All munitions described above are modeled.

Side	Unit Type	Quantity
BLUE	MCS	3
	ICV	3
	SUAV	5
	NLOS LS	4
	Mortar	4
	ARV	3
RED	MBT	12
	ICV	8
	Jeep	6

Table 5. Analysis Scenario Unit Compositions

1. Factors

As an initial effort in using DAFS for analysis, a three-factor experiment was developed to demonstrate the capability of DAFS to provide usable output. The factors chosen were tactical values, BDA factor and optimization interval. The tactical values were altered between two states. The first having all units assigned the same tactical values and the second having blue tactical values set at half the red. Red tactical values less than blue would not provided allocated engagement assignments due to the Value of Potential Assignments (VPA) instance used. The second factor, BDA factor, is varied in three values. (0.5, 0.7, 0.9) These values represent the expected percentage of accurate BDA reports. The final factor, optimization interval, is varied at 0.75 and 1.25 hours and applies to both the fires allocation CVO and the BDA CVO.

2. Scenario Replication

The factors described above are varied one at a time over the full twelve possible combinations. At each factor combination, 40 replications of the first twelve hours of combat are modeled. For all twelve-factor settings, this represents 240 full combat days. As the factors are varied over the full combinatorial pattern, the survivability rates of the blue and red units are measured at the end of each replication.

3. Discussion of Results

The simulation results were analyzed using a linear regression model to determine the significance of each factor in the survivability rates for blue and red. The models generated are separate for blue and red survivability and the technical output is presented in Appendix C. Interaction models were also generated but showed that no significant affects could be attributed to the interaction of factors.

In general it can be stated that all three factors explored are significant to the simulation results. Specifically, all three factors are significant to the level of blue survivability and the BDA factor is significant to red survivability.

In the case of blue, the p significance values for tactical value, BDA factor and optimization interval are 3.5×10^{-3}, 2.7×10^{-3} and 6.7×10^{-3} respectively. This indicates that in the presence of the other factors, each factor was significant in the ability to predict the mean value of the blue survival rate. However, the R^2 value for the blue survivability model was 5.0×10^{-2}, indicating that very little of the variance in survivability was explained by the linear model.

In the case of red, the p significance values for tactical value, BDA factor and optimization interval are 1.41×10^{-1}, 0.0 and 2.89×10^{-1} respectively. The indication here is that only the BDA factor was significant to the prediction capability of the model and that the other factors, in the presence of the BDA factor, were not significant. The R^2 value for red was 2.77×10^{-1}, which while not a strong value, does mean that more of the variance in red survivability was explained in the linear model

In summation, both linear models demonstrate that the factors chosen for this analysis are significant to the output. While this analysis does not capture the depth of considerations required to be analyzed for full consideration of networked assets, the following general statements can be made:

- ❖ Tactical value, BDA factor and optimization interval are significant in the performance of the forces modeled with respect to the survivability.
- ❖ Based on a consideration of both R^2 values, the analysis indicates that the blue force survivability is more robust to the changing of factors and means that policies, based on these factors alone, could be set with significantly more weight applied to red survivability.
- ❖ The DAFS framework is capable of providing relevant analysis output in the evaluation of networked fires and FCS unit configurations.

4. Additional Analysis

Upon completion of the analysis runs just discussed, DAFS was exercised under increasingly large scenarios to determine limiting factors. For the full model described in section A of this chapter, DAFS completed the twelve hour scenario in six hours; a simulation pitting 98 red force units against 78 blue units (including 32 SUAV's). To this point DAFS has successfully simulated scenarios containing 93 blue units and 120 red units without surpassing its capabilities. These runs have taken between 5 and 9 hours to complete.

VI. CONCLUSIONS

A. CONCLUSIONS

The Dynamic Allocation of Fires and Sensors (DAFS) model has demonstrated its ability to be used for analyzing both networking of assets and Future Combat System (FCS) Entity and Unit of Action (UA) configurations. The results reached in the previous chapter are examples of the research capabilities supportable through DAFS.

The DAFS model framework offers several significant benefits in this area of research and has initially proven its applicability. The following contributions are specifically noted:

- DAFS is an extremely flexible analysis framework. The component based nature of the model offers the user the capability to modify and/or generate new versions of several components and conduct additional evaluations. The following interface based components demonstrate this flexibility:
    - Constrained Value Optimizer (CVO). Different LP formulations may be devised to test the allocation process.
    - Value Of Potential Assignments (VPA) module. The logic used to provide the objective function constraints to the CVO may be altered or changed in combination with or separate from CVO changes. Changes to the VPA also include considerations of different factors in the munitions/sensors allocation logic.
    - Platform. New platforms may be tested with different operating parameters and configured with the following elements that may also be changed:
        - Munitions
        - Mover managers
        - Sensors

- Kill Probabilities. The XML file inputs that define the predicted kill probabilities as a function of munitions type and platform type may be modified to include more complicated functions.
- Command. The prioritization and assignment handling logic in the command element controlling the platform may be altered.

❖ DAFS scenarios are easily configurable, which contributes to the potential for more expeditious analysis.

❖ The DAFS model is a timely foundation for the analysis of networked fires and FCS unit configurations; meeting the Army's need to understand the factors involved in the fielding of the Objective Force.

B. RECOMMENDATIONS

This author strongly recommends the continued pursuit of research in networked fires and other FCS considerations using DAFS. An expansion of the scope of DAFS and further refinement of the logic applied may lead to beneficial insights as TRAC-Monterey and the Army draw near to milestones in the Objective Force development process. Additionally, DAFS is especially suitable for partnering with other efforts in the same research area.

Expansion of the scope of considerations modeled in DAFS is necessary to achieve a higher level of confidence and demonstrable level of validity. Specifically covered in the following section, there are several areas in which DAFS can be augmented to allow better resolution and fidelity in the analysis of networked fires and FCS configurations. It has been a sincere focus of this thesis to ensure the capability of augmentation exists and that the flexibility of DAFS truly allow more extended application than was permitted in the time frame of this work.

DAFS is also very well suited for partnering with parallel efforts. As a test bed, DAFS can assist other research efforts that do not entail the actual simulation of results or that are largely theoretical. Results achieved through similar or other means may be tested within DAFS and may either be used for comparison or in the case of theoretical or deterministic studies, be used to validate the results.

By focusing additional research efforts through DAFS and partnering DAFS capabilities with other efforts being undertaken throughout all TRAC commands, measurable benefits may be achieved. DAFS provides an extremely flexible, easily configurable analysis framework that may lend a great deal of foundation for significant decisions being made by the Army in the coming years.

C. FOLLOW ON RESEARCH

Because this thesis represents the ground-level efforts for an larger project being taken on by the Army and TRAC-Monterey, the opportunities for follow on research are many. Continued efforts in this project are available in both operations research and modeling and the conceptual framework is largely in place. However, for the Army to truly benefit from this model, enhancements are required in the model itself and the complexity of the operations research applications need to be explored and expanded.

Throughout the process of conducting this research, several specific items became apparent as potentially beneficial to the capability of the DAFS model. These items fall into the two general categories. The first is Operations Research (OR). These areas focus on the design of experiments and factor analysis of topics being considered and on the specifics of the OR disciplines used. The second area is more related to the functionality of the model and deals with potential improvements in human interface and expandability of functions.

1. Operations Research

This thesis has merely scraped the surface of potential for analysis in networked fires and FCS configurations. The following is a list of some of the areas in which DAFS may be utilized in additional OR related research.

- Alter or expand the considerations made in the CVO and VPA.

- ❖ Utilize DAFS asset allocation format to study other factors such as logistics or aggregated units.
- ❖ Analyze the potential for oscillating allocation decisions based on opponent forces use of similar tactics and potential methods to overcome such a situation.
- ❖ Expand the factor space for consideration and explore the use of robust design using DAFS.
- ❖ Conduct analysis on a range of scenarios utilizing steady configurations and optimization logic to explore the robustness of potential decisions.
- ❖ Increase the level of "fog" associated with information in the battle space and include additional stochastic events such as hardware failures, contact identification and contact correlation.
- ❖ Apply the results of preliminary research to explore the potential for dynamic optimization control including but not limited to constraint values, optional VPA implementations and variable optimization intervals.

2. Model Functionality

Continued effort in the development of DAFS regarding ease of use may be very beneficial. The following are some of the potential considerations

- ❖ Expand the capability of output definition as a result of input.
- ❖ Develop and implement XML support documents including but not limited to Schemas, DTDs, and XSL documents for output manipulation and external source data conversion. This may be particularly beneficial for cross analysis with other models.
- ❖ Develop a graphical user interface that allows scenario definition completely based on data files and driven by menus including the

option for discrete unit construction or default entities. Additionally, the interface may be developed to allow XML data files to be written upon scenario development or at an interruption point in the simulation that may then be re-loaded.

- Extend the capability of the simulation to account for terrain and object interferences. This is particularly critical to the Army for eventual analysis of FCS unit configuration and employment in the urban environment.

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

Ahmed, Kal, Sudhir Ancha, Andrei Cioroianu, Jay Cousins, Jeremy Crosbie, John Davies, Kyle Gabhart, Steve Gould, Ramnivas Laddad, Sing Li, Brendan Macmillan, Daniel Rivers-Moore, Judy Skubal, Karli Watson, Scott Williams and James Hart, Professional JAVA XML, Wrox press Ltd, Birmingham, UK: 2001.

Buss, Arnold, "Component Based Simulation Modeling", Proceedings of the 2000 Winter Simulation Conference, J.A. Joines, R.R. Barton, K. Kang and P.A. Fishwich, eds, 2000.

Buss, Arnold and Paul Sanchez, "Building Complex Models with LEGOS (Listener Event Graph Objects)", Proceedings of the 2002 Winter Simulation Conference, E. Yucesan, C.H. Chen, J.L. Snowden, and J.M. Diarmes, eds, 2002.

Jackson, Jack and Arnold Buss, Standard Army Modeling and Simulation Objects (SAMSO), U.S. Army TRADOC Analysis Center, Fort Leavenworth, KS: 1998.

Joint Vision 2020, U.S. Government Printing Office, Washington D.C.: 2000.

TRADOC PAM 525-3-90/O&O, U.S. Army Training and Doctrine Command, Fort Monroe, VA: 2002.

Law, Averil M. and David W. Kelton, Simulation Modeling and Analysis, 3rd edition, McGraw Hill, 2000.

Matsumura, John, Randall Steeb, Tom Herbert, John Gordon, Carl Rhodes, Russell Glenn, Michael Barbero, Fred Gellert, Phyllis Kantar, Gail Halverson, Robert Cochran and Paul Steinberg, Exploring Advanced Technologies for the Future Combat Systems Program, RAND, Santa Monica, CA: 2002.

Ragsdale, Cliff T., Spreadsheet Modeling and Decision Analysis, 3rd edition, South-Western Publishing, Cincinnati, OH: 2001.

Rardin, Ronald L., Optimization in Operations Research, Prentice Hall, New Jersey: 1998

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX A: DYNAMIC ALLOCATION OF FIRES AND SENSORS SPREADSHEET TOOL (DAFS-ST)

A. SPREADSHEET FUNCTION

The function of the spreadsheet used with the DAFS model is to provide a tool for initial analysis of factors and algorithms associated with assigning fires responsibilities throughout the participating force. The assignment of fires responsibility is not in itself an order to fire. Rather, it is an assignment of responsibility. When this logic is applied to the DAFS simulation it allows for prioritization of unit actions. The basic functions of the full workbook are described here followed by more detailed descriptions of each worksheet.

1. Battle Space Configuration

User defines players. From 0 to 10 tanks, jeeps and Armored vehicles for each side. The type of unit is actually insignificant; the names merely serve as an identifier for units with different capabilities.

User defines operating areas. Each type of player can be limited to areas on the battlefield defined by Cartesian rectangles.

User defines constraint thresholds. Scenario type, minimum and maximum assignments per blue unit, by type; highest acceptable threat to own unit, and lowest desired probability of kill are all factors that the user may set that are subsequently used as constraints.

Positions generated. The players are *randomly* placed on the battlefield within the areas designated using the reconfigure button on the generate page. As an option, the user may go directly to the locations page and enter positions for each element.

2. Initial Calculations

Tables. All relative information is calculated immediately after the positions are set. These calculations populate tables for all pair-wise ranges (blue to red), the associated Pk's and the associated threat values. At this point, binary decision multipliers associated with Pk and threat, are also generated as a result of comparison with the entered limits discussed earlier. These multipliers drive the value to negative one if the Pk or threat values do not meet the user constraints.

Assignment Value. The potential value of each assignment is calculated using the previously generated tables and additional tabular data (unit values). The assignment value formula currently used is explained in the pairings worksheet section below.

3. Solver Generated Pairings

Solver set-up. The varying cells, and the associated binary constraints must be set-up consistent with the generated players. Constraints for maximum blue assignments and the minimum and maximum red coverage constraints do not need to be adjusted after the first set-up. The configuration of these constraints is sufficiently generic to support all player configurations. Also, multiple runs may be accomplished with the same players in different positions without requiring additional solver set-up.

Data save. The results of any particular run can be copied to a history worksheet for further analysis.

B. WORKSHEET DESCRIPTIONS

1. Input

Provides the input interface for constraints and players, and the final output image.

Players. 0 –10 tanks, jeeps and AV's for each side.

Area Assignments. Maximum and minimum X and Y coordinates for each color and type of unit. This cannot be done at the individual entity level.

Scenario mission. Peacekeeping, attack or defend; this option influences player inherent values.

Lowest desired Pk. Minimum probability of kill from a blue to red that will allow a potential assignment to be considered.

Maximum threat. Maximum threat Pk from red to blue acceptable in a potential assignment.

Cover all targets. Yes sets a constraint forcing at least one blue unit to be assigned to each red unit. (User must be careful not to define an infeasible problem)

Max assignments. Limits the number of red units that can be assigned to a particular blue unit. Assigned by type.

Tabulate. Clears the pairings displayed and recalculates tables. This function is used when parameters are changed and players or areas are not.

Copy. Copies comments, parameters, positions, assignment matrix and the battlefield image to the copies page for data recording.

2. Generator

Randomly generates locations for all *possible* players and a participation indicator. Provides the user the opportunity to select the random configuration desired for analysis.

Locations. Each side has 30 permanent members, 10 of each type. On reconfigure, new locations for each are generated randomly conforming to the area constraints from the inputs page. Additionally, the participation indicator is calculated based on the number of each type of player dictated on the inputs page. The location coordinates are then multiplied by the participation indicator. This results in positive values only for those players desired.

Participation indicator. 1 or –1. This value is set to 1 if a player is to be used and –1 if not. This logic indicator is used throughout the spreadsheet to control plotting and calculations.

Use this**.** The "use this" button copies the positions associated with the displayed scenario to the locations page for further application.

3. Locations

This page contains the working positions for a particular set of analysis runs. Additionally, the user may paste values here from the copies page and re-evaluate a run accomplished in the past, perhaps with different input parameters.

4. Tables

These data tables are filled with proxy data designed to capture the essence of diminishing Pk as a function of range (Based on professional military judgement). These values also capture some characteristic nature of the potential effectiveness of a unit against a particular target type. For example, the effectiveness of a jeep against a tank is far less than the reverse.

Pk values. Based on range, these are the probability of kill values from blue to red.

Threat values. Also based on range, these represent the probability of kill from red to blue.

Player values. Qualitative inherent values for each type of player based on the mission.

Position factors. Intended as a weighting factor in the consideration for move to engage situations. Based on the mission, this factor is not currently used.

5. Calculations

Contains 5 tables that are calculated based on active possible pairings. Specifically, each block in the tables represents the appropriate value for an interaction between two unique players. The calculations are only made if both players have participation indicators of 1.

Ranges. All pair wise ranges. Unit-less.

Pk values. The Pk from blue to red associated with the potential pairing as a function of range.

Threat values. Same as Pk values but from red to blue.

Pk and Threat multipliers. 1 or 0 based on whether or not the corresponding value in the Pk or threat tables meets the acceptability limit from the inputs page. Used in the valuation formula, these multipliers drive the value of the associated assignment value to negative one if the desired constraints are not met.

6. Pairings

The binary decision matrix and assignment value matrix are on this sheet along with the pairings location matrix.

Pairings. Binary decision matrix. This is the table used by solver to set pairings. Row sums are compared to the max assignment constraint and column sums are compared to the cover all and max assigned to target constraints.

Assignment values. Value of the particular assignment as a function of previously calculated tables. Currently this value is calculated as follows.

Value = [RedValue*Pk – BlueValue*(1-threat)]*PkMult*threatMult

where:

RedValue*Pk represents the expected benefit of the assignment from red damage.

BlueValue*(1-threat) represents the expected value of the assignment from remaining blue capability.

PkMult as described earlier, drives the value to 0 if desired minimum Pk constraint is not met.

ThreatMult serves the same function as the PkMult for the threat constraint.

Pairings location matrix. Mirrors the assignment matrix only the contents are the pair wise locations of the blue and red units. Used for plotting the selected pairing lines on the inputs page.

7. Copies

As described earlier, this page is recorded analysis runs. The parameter table, parings matrix, locations of all units and the battlefield snapshot are recorded. From this page, the analysis may be completely reset and analyzed differently. This is vary beneficial to the project.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B: SAMPLE XML FILES

The Following are portions of sample files used with DAFS and are representative of the full files used.

1. KILL PROBABILITIES


```
<?xml version="1.0" encoding="UTF-8"?>
<KillProbabilities>
    <KillProbability type="dafs.LinearKillProbability" munitionType="PAM"
          platformType="Netfires">
        <Params minRange="0.5" maxRange="50" maxRangePK="0.98"
          minRangePK="0.98"></Params>
    </KillProbability>
    <KillProbability type="dafs.LinearKillProbability" munitionType="PAM"
       platformType="RedAPC">
        <Params minRange="0.5" maxRange="50" maxRangePK="0.92"
          minRangePK="0.92"></Params>
    </KillProbability>
    <KillProbability type="dafs.LinearKillProbability" munitionType="PAM"
          platformType="RedTank">
        <Params minRange="0.5" maxRange="50" maxRangePK="0.9"
          minRangePK="0.9"></Params>
    </KillProbability>
    <KillProbability type="dafs.LinearKillProbability" munitionType="PAM"
          platformType="Mortar">
        <Params minRange="0.5" maxRange="50" maxRangePK="0.98"
          minRangePK="0.98"></Params>
    </KillProbability>
    <KillProbability type="dafs.LinearKillProbability" munitionType="PAM"
          platformType="IFV">
        <Params minRange="0.5" maxRange="50" maxRangePK="0.98"
          minRangePK="0.98"></Params>
    </KillProbability>
    <KillProbability type="dafs.LinearKillProbability" munitionType="PAM"
          platformType="Tank">
        <Params minRange="0.5" maxRange="50" maxRangePK="0.98"
          minRangePK="0.98"></Params>
    </KillProbability>
    <KillProbability type="dafs.LinearKillProbability" munitionType="PAM"
          platformType="Recon">
        <Params minRange="0.5" maxRange="50" maxRangePK="0.98"
          minRangePK="0.98"></Params>
    </KillProbability>
    <KillProbability type="dafs.LinearKillProbability" munitionType="PAM"
          platformType="RedJeep">
        <Params minRange="0.5" maxRange="50" maxRangePK="0.98"
          minRangePK="0.98"></Params>
    </KillProbability>
    <KillProbability type="dafs.LinearKillProbability" munitionType="OCSW"
          platformType="Netfires">
        <Params minRange="0" maxRange="2" maxRangePK=".25"
          minRangePK=".8"></Params>
    </KillProbability>
</KillProbabilities>
```

2. BASE SCENARIO FILE


```
<?xml version="1.0" encoding="UTF-8"?>
<DafsScenario type="Attack" optimizeInterval="1" bdaFactor=".95"
       replications="20">
    <SimEntity>
       <Mover qty="3" type="dafs.Platform" affiliation="Blue" platform="Tank"
              assignment="fires">
          <Parameters maxSpeed="25.0"></Parameters>
          <Position>
             <Grid xLoc="0" yLoc="0"></Grid>
          </Position>
          <Sensor type="dafs.DAFSSensor" sensor="Radar">
             <Parameters name="maxRange" value="4"></Parameters>
          </Sensor>
          <MoverManager type="dafs.DAFSPathMoverManager">
                <WPBox minX="-3" maxX="3" minY="-3" maxY="3" order="1"></WPBox>
          </MoverManager>
          <Munitions>
              <Munition munitionType="CKEM">40</Munition>
              <Munition munitionType="OCSW">1800</Munition>
              <Munition munitionType="LCPK">40</Munition>
              <Munition munitionType="MedCal">1200</Munition>
          </Munitions>
          <Box minX="-20" maxX="-19" minY="-5" maxY="5">
          </Box>
       </Mover><Mover qty="2" type="dafs.Platform" affiliation="Blue"
              platform="Netfires" assignment="fires">
          <Parameters maxSpeed="20.0"></Parameters>
          <Position>
             <Grid xLoc="0" yLoc="0"></Grid>
          </Position>
            <Sensor type="dafs.DAFSSensor" sensor="Radar">
             <Parameters name="maxRange" value="4"></Parameters>
          </Sensor>
          <MoverManager type="dafs.DAFSPathMoverManager">
          </MoverManager>
          <Munitions>
              <Munition munitionType="PAM">15</Munition>
              <Munition munitionType="OCSW">1800</Munition>
          </Munitions>
          <Box minX="-36" maxX="-35" minY="25" maxY="26"></Box>

       <Mover qty="1" type="dafs.Platform" affiliation="Blue" platform="SUAV"
              assignment="sensor">
          <Parameters maxSpeed="50.0"></Parameters>
          <Position>
             <Grid xLoc="0" yLoc="0"></Grid>
          </Position>
            <Sensor type="dafs.DAFSSensor" sensor="Radar">
             <Parameters name="maxRange" value="10"></Parameters>
          </Sensor>
          <MoverManager type="dafs.DAFSPatrolMoverManager">
            <WPBox minX="10" maxX="20" minY="0" maxY="20" order="1"
                 qty="8"></WPBox>
          </MoverManager>
          <Box minX="-1" maxX="1" minY="20" maxY="21"></Box>
```

```
        <Mover qty="4" type="dafs.Platform" affiliation="Red"
             platform="RedAPC">
          <Parameters maxSpeed="25.0"></Parameters>
          <Position>
             <Grid xLoc="100.0" yLoc="100.0"></Grid>
          </Position>
          <Sensor type="dafs.DAFSSensor" sensor="Radar">
             <Parameters name="maxRange" value="5"></Parameters>
          </Sensor>
          <MoverManager type="dafs.DAFSPathMoverManager">
             <WPBox minX="14" maxX="18" minY="-20" maxY="-10" order="1"
                 qty="1"></WPBox>
          </MoverManager>
          <Munitions>
             <Munition munitionType="RedJavelin">6</Munition>
             <Munition munitionType="RedOCSW">1800</Munition>
          </Munitions>
          <Box minX="22" maxX="26" minY="-25" maxY="-15"></Box>
        </Mover>
        <Mover qty="2" type="dafs.Platform" affiliation="Red"
             platform="RedJeep">
          <Parameters maxSpeed="30.0"></Parameters>
          <Position>
             <Grid xLoc="100.0" yLoc="100.0"></Grid>
          </Position>
          <Sensor type="dafs.DAFSSensor" sensor="Radar">
             <Parameters name="maxRange" value="9"></Parameters>
          </Sensor>
          <MoverManager type="dafs.DAFSPathMoverManager">
                <WPBox minX="-2" maxX="-1" minY="-3" maxY="3" order="1"
                    qty="5"></WPBox>
          </MoverManager>
          <Munitions>
             <Munition munitionType="RedJavelin">6</Munition>
             <Munition munitionType="RedMedCal">1200</Munition>
          </Munitions>
          <Box minX="-2" maxX="-1" minY="-3" maxY="3"></Box>
        </Mover>
   <Constraints maxAssign="2" maxCover="2" cover="false" minPK="0.7"
       maxThreatPK="1.0" minCover="0"></Constraints>
</DafsScenario>
```

3. PLATFORM VALUES


```xml
<?xml version="1.0" encoding="UTF-8"?>
<PlatformValues>
    <ScenarioValues scenarioType="Attack">
        <Value platformType="Tank">2000</Value>
        <Value platformType="RedTank">2000</Value>
        <Value platformType="IFV">2000</Value>
        <Value platformType="Recon">2000</Value>
        <Value platformType="SUAV">600</Value>
        <Value platformType="Netfires">2000</Value>
        <Value platformType="Mortar">2000</Value>
        <Value platformType="RedAPC">2000</Value>
        <Value platformType="RedJeep">2000</Value>
    </ScenarioValues>
    <ScenarioValues scenarioType="Defend">
        <Value platformType="Tank">1000</Value>
        <Value platformType="RedTank">1000</Value>
        <Value platformType="IFV">1000</Value>
        <Value platformType="Recon">1000</Value>
        <Value platformType="SUAV">600</Value>
        <Value platformType="Netfires">1000</Value>
        <Value platformType="Mortar">1000</Value>
        <Value platformType="RedAPC">2000</Value>
        <Value platformType="RedJeep">2000</Value>
    </ScenarioValues>
</PlatformValues>
```

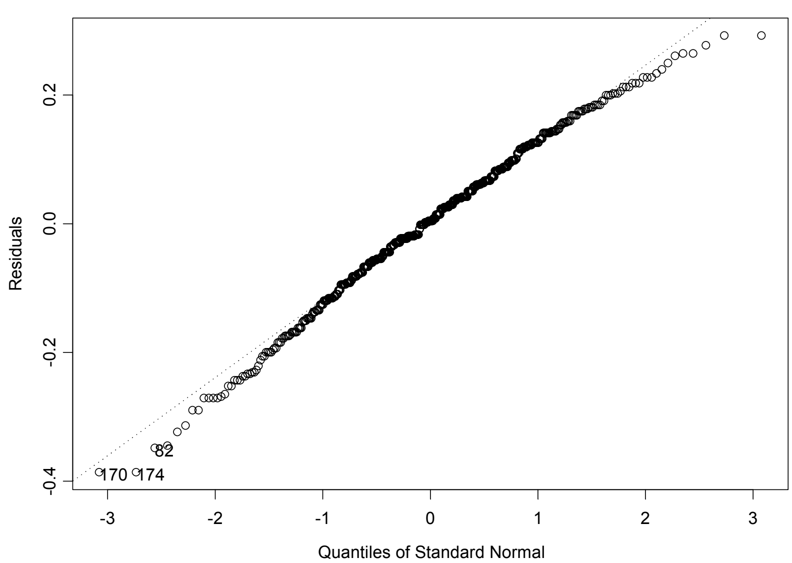

4. SIMULATION RUNNER

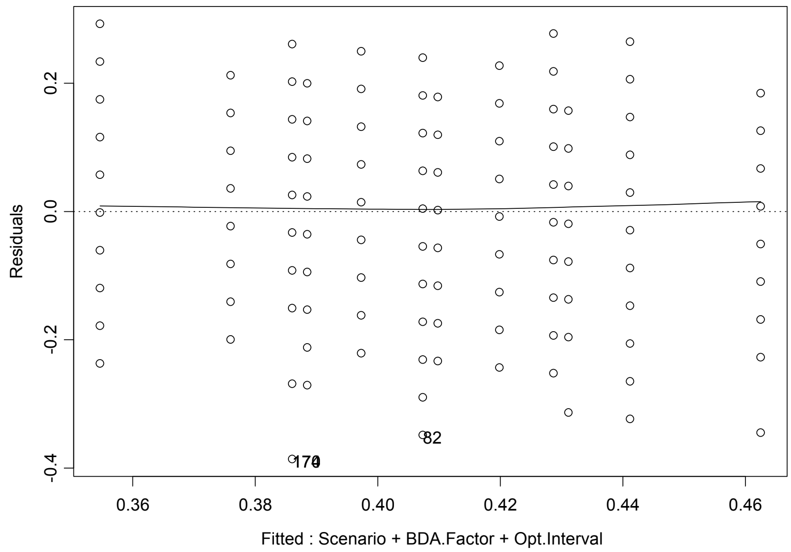

```xml
<?xml version="1.0" encoding="UTF-8"?>
<run>
    <StopType>
        <StopAtTime>
            <stopTime>30</stopTime>
        </StopAtTime>
    </StopType><Verbose>false</Verbose>
    <SingleStep>false</SingleStep>
    <NumberReplications>3</NumberReplications>
</run>
```

APPENDIX C: REGRESSION RESULTS

1. BLUE SURVIVABILITY VS. FACTORS

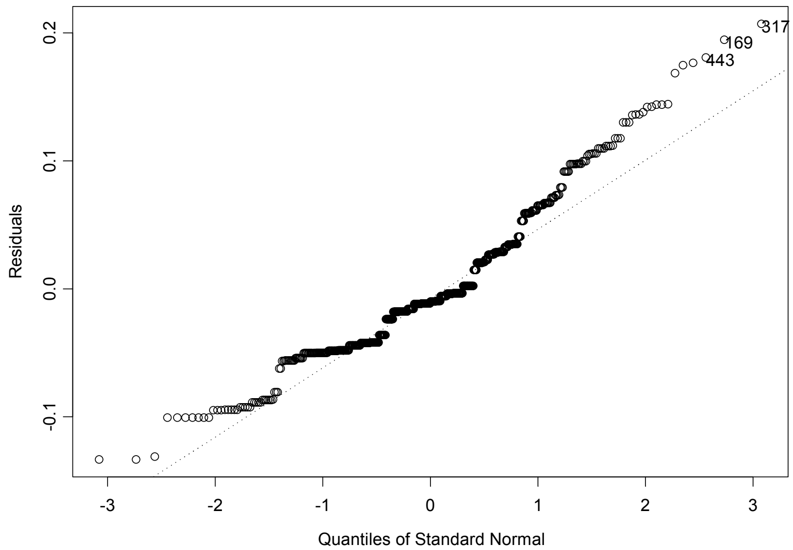
```
        *** Linear Model ***

Call: lm(formula = Blue.Surv ~ Scenario + BDA.Factor + Opt.Interval, data
= out.data, na.action = na.exclude)
Residuals:
    Min       1Q   Median      3Q    Max
 -0.386 -0.07819 0.004412 0.08548 0.2924

Coefficients:
                Value Std. Error  t value Pr(>|t|)
 (Intercept)   0.4086   0.0058    70.9015   0.0000
    Scenario   0.0169   0.0058     2.9347   0.0035
  BDA.Factor  -0.0213   0.0071    -3.0213   0.0027
Opt.Interval  -0.0157   0.0058    -2.7221   0.0067

Residual standard error: 0.1263 on 476 degrees of freedom
Multiple R-Squared: 0.05019
F-statistic: 8.384 on 3 and 476 degrees of freedom, the p-value is
0.00001932

Analysis of Variance Table

Response: Blue.Surv

Terms added sequentially (first to last)
              Df Sum of Sq   Mean Sq  F Value       Pr(F)
    Scenario   1  0.137284 0.1372837 8.612675 0.003499667
  BDA.Factor   1  0.145502 0.1455017 9.128242 0.002652446
Opt.Interval   1  0.118108 0.1181084 7.409687 0.006725338
   Residuals 476  7.587313 0.0159397
```



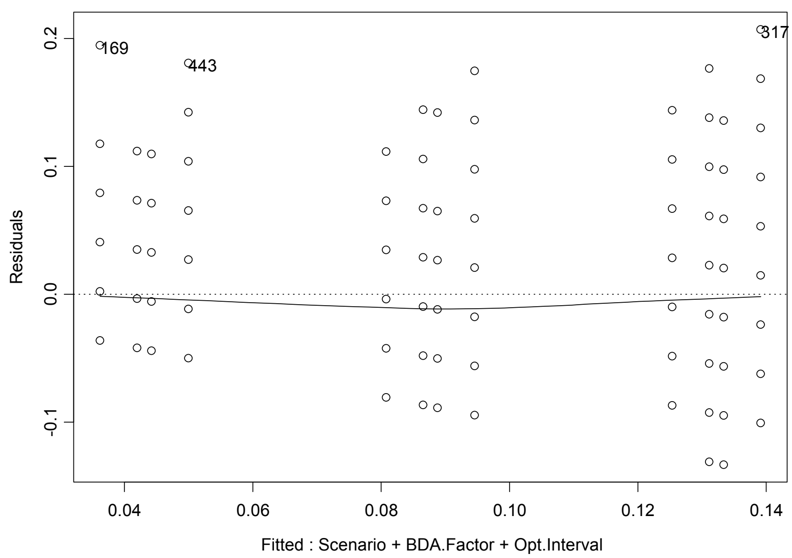


2. RED SURVIVABILITY VS. FACTORS

```
        *** Linear Model ***

      Call: lm(formula = Red.Surv ~ Scenario + BDA.Factor + Opt.Interval, data
= out.data, na.action = na.exclude)
      Residuals:
          Min       1Q   Median      3Q   Max
       -0.1334 -0.04419 -0.01074 0.02885 0.207

      Coefficients:
                      Value Std. Error  t value Pr(>|t|)
       (Intercept)   0.0877   0.0027    32.2301   0.0000
          Scenario   0.0040   0.0027     1.4730   0.1414
        BDA.Factor  -0.0446   0.0033   -13.3864   0.0000
      Opt.Interval   0.0029   0.0027     1.0606   0.2894

      Residual standard error: 0.05959 on 476 degrees of freedom
      Multiple R-Squared: 0.2771
      F-statistic: 60.83 on 3 and 476 degrees of freedom, the p-value is 0

      Analysis of Variance Table

      Response: Red.Surv

      Terms added sequentially (first to last)
                    Df Sum of Sq   Mean Sq  F Value     Pr(F)
          Scenario   1  0.007705 0.0077046   2.1698 0.1414014
        BDA.Factor   1  0.636284 0.6362842 179.1957 0.0000000
      Opt.Interval   1  0.003994 0.0039941   1.1248 0.2894150
         Residuals 476  1.690170 0.0035508
```



317
169
443
Residuals
Quantiles of Standard Normal



169
443
317
Residuals
Fitted : Scenario + BDA.Factor + Opt.Interval

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. Professor Arnold H. Buss
   Naval Postgraduate School
   Monterey, California

4. Professor Paul Sanchez
   Naval Postgraduate School
   Monterey, California

5. Professor Don Gaver
   Naval Postgraduate School
   Monterey, California

6. TRADOC ANALYSIS CENTER
   ATTN ATRC (Mr. Bauman) Fort Leavenworth Kansas

7. Director
   TRADOC Analysis Center - Monterey
   Naval Postgraduate School
   Monterey California

8. HQDA, DCS G3, (DAMO-ZS
   ATTN: LTC Wilson
   Washington DC

9. TRADOC Analysis Center – Monterey
   ATTN: MAJ Chesney
   Naval Postgraduate School
   Monterey, California

10. Paul Hedeman
    BAE Systems
    Information and Electronic Warfare Systems
    Nasua, New Hampshire